(12)特許協力条約に基づいて公開された国際出願

551 391

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2004 年10 月14 日 (14.10.2004)

PCT

(10) 国際公開番号
WO 2004/088500 A1

(51) 国際特許分類[7]: G06F 7/00

(21) 国際出願番号: PCT/JP2004/004752

(22) 国際出願日: 2004 年3 月31 日 (31.03.2004)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:

特願2003-093922	2003 年3 月31 日 (31.03.2003)	JP
特願2003-105762	2003 年4 月9 日 (09.04.2003)	JP

(71) 出願人 (米国を除く全ての指定国について): 財団法人北九州産業学術推進機構 (KITAKYUSHU FOUNDATION FOR THE ADVANCEMENT OF INDUSTRY, SCIENCE AND TECHNOLOGY) [JP/JP]; 〒8080135 福岡県北九州市若松区ひびきの２－１ Fukuoka (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 笹尾 勤 (SASAO, Tsutomu) [JP/JP]; 〒8100053 福岡県福岡市中央区鳥飼３丁目１５－１７ Fukuoka (JP). 井口 幸洋 (IGUCHI, Yukihiro) [JP/JP]; 〒2500408 神奈川県足柄下郡箱根町強羅１３２０－１２３１ Kanagawa (JP).

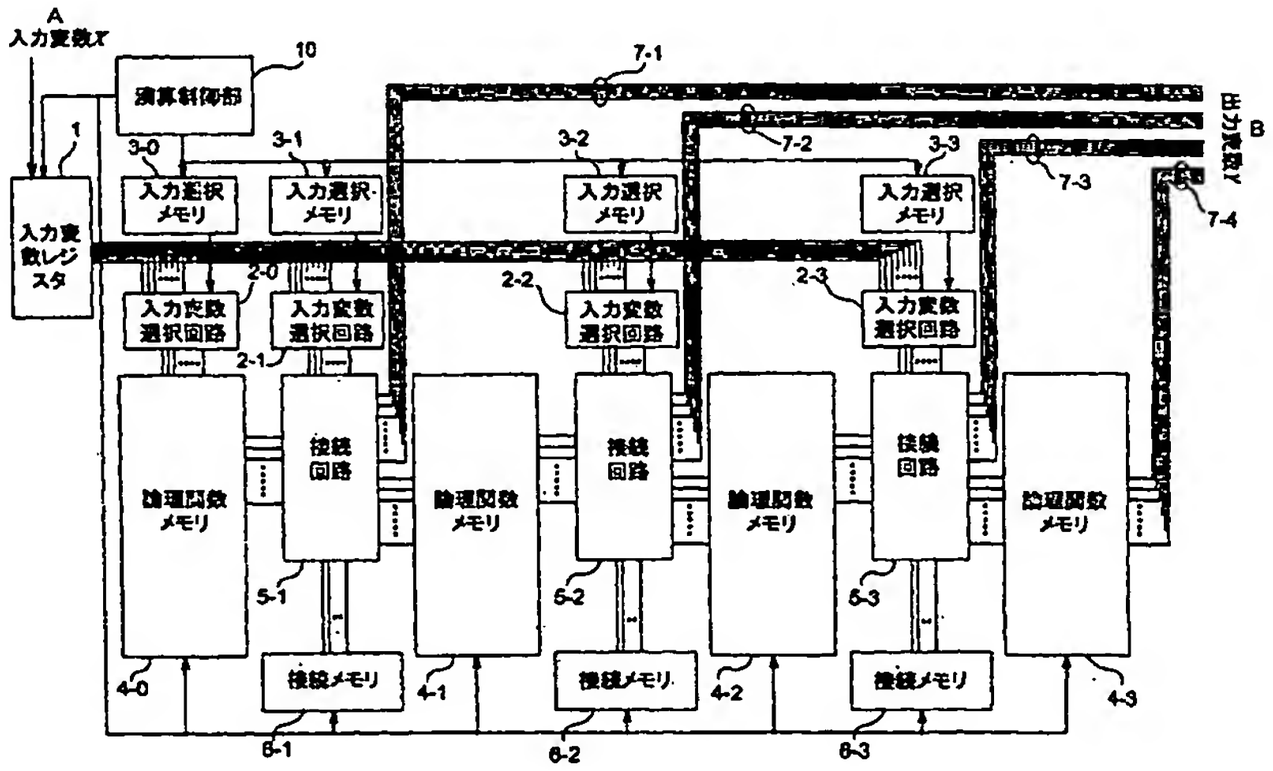
(74) 代理人: 石田 和人 (ISHIDA, Kazuto); 〒8080135 福岡県北九州市若松区ひびきの２－１ 北九州学術研究都市産学連携センター T-302 Fukuoka (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE,

[続葉有]

(54) Title: PROGRAMMABLE LOGIC DEVICE

(54) 発明の名称: プログラマブル論理デバイス



A...INPUT VARIABLE X
B...OUTPUT VARIABLE Y
1...INPUT VARIABLE REGISTER
10...OPERATION CONTROL SECTION
3-0...INPUT SELECTION MEMORY
3-1...INPUT SELECTION MEMORY
2-0...INPUT VARIABLE SELECTION CIRCUIT
2-1...INPUT VARIABLE SELECTION CIRCUIT
4-0...LOGIC FUNCTION MEMORY
5-1...CONNECTION CIRCUIT
6-1...CONNECTION MEMORY
3-2...INPUT SELECTION MEMORY
2-2...INPUT VARIABLE SELECTION CIRCUIT
4-1...LOGIC FUNCTION MEMORY
5-2...CONNECTION CIRCUIT
6-2...CONNECTION MEMORY
3-3...INPUT SELECTION MEMORY
2-3...INPUT VARIABLE SELECTION CIRCUIT
4-2...LOGIC FUNCTION MEMORY
5-3...CONNECTION CIRCUIT
6-3...CONNECTION MEMORY
4-3...LOGIC FUNCTION MEMORY

(57) Abstract: There is provided a programmable logic device capable of performing optimal designing by changing the number of input lines between the logic function memories and the number of rails and suppressing the memory capacity to a minimum value according to the target logic function. The logic function memories (4) are successively connected in series and store LUT. An input variable is input to each of the logic function memories (4) from an external

[続葉有]

WO 2004/088500 A1

SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告書

— 請求の範囲の補正の期限前の公開であり、補正書受領の際には再公開される。

２文字コード及び他の略語については、定期発行される各*PCT*ガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

input line. Between two logic function memories (4), according to the connection information stored in a connection memory (6), a connection circuit (5) connects an output line or an external input line of the logical function memory (4) of a former stage to an input line of the logical function memory (4) of a latter stage. By rewriting the connection information in accordance with the target logic function, it is possible to reconstruct the connection circuit and change the number of input lines and the number of rails. It is possible to optimize the ratio of the number of rails against the number of the input lines in accordance with the logic function and suppress the memory capacity to a minimum value.

(57) 要約: 目的論理関数に応じ、論理関数メモリ間の入力線数、レイル数を変化させ、メモリ容量を必要最小限に抑えて最適化設計できるプログラマブル論理デバイスである。論理関数メモリ（４）を直列に順序配列し、ＬＵＴを記憶させる。外部入力線から、各論理関数メモリ（４）への入力変数を入力する。二つの論理関数メモリ（４）間において、接続メモリ（６）が記憶する接続情報に従って、接続回路（５）が前段の論理関数メモリ（４）の出力線又は外部入力線と後段の論理関数メモリ（４）の入力線との接続を行う。目的論理関数に合わせて、接続情報を書き換えれば、接続回路を再構成し、入力線数、レイル数を変化させることができる。レイル数と入力線数の比を論理関数に合わせて最適化し、メモリ容量を必要最小限に抑えることが可能となる。

明　細　書

プログラマブル論理デバイス

技術分野

本発明は、プログラム可能な論理デバイス（programmable logic device）に係り、特に、目的論理関数を、順序づけられた複数の分解関数に関数分解し、それら分解関数を分解表として表現したルックアップ・テーブル・カスケード（look up table cascade；以下「ＬＵＴカスケード」という。）の演算を行うためのプログラマブル論理デバイスに関する。

背景技術

近年、論理回路の設計においては、フィールド・プログラマブル・ゲートアレイ（Field Programmable Gate Array。以下、「ＦＰＧＡ」という。）が広く用いられるようになっている（引用文献［１］）。ＦＰＧＡは、マトリックス状に配列された複数の論理セル（Configurable Logic Block；以下、「ＣＬＢ」という。）の内容と論理セル間の配線の接続とをメモリの書き換えによって変更することが可能なプログラマブル論理デバイスである。ＦＰＧＡは、プログラムに従ってソフトウェア的に演算を行うマイクロ・プロセッサ（Micro Processing Unit；以下、「ＭＰＵ」という。）とは異なり、ハードウェア的に演算を行う。従って、論理関数の演算実行速度が速いという特徴を有する。

一方、ＦＰＧＡは、各ＣＬＢ間の物理的な配線の引き回しをプログラムにより変更する。そのため、配線遅延をできる限り小さくするように考慮した配置配線設計が必要とされ、設計に長時間を要する。また、各ＣＬＢ間の配線の引き回し方に依存して

、配線遅延時間が変化する。そのため、設計時において論理回路の演算時間の予測が困難である。

また、２次元的に配列されたＣＬＢ間の配線を、再構成可能なように自在に接続切り替えを行うことから、チップ内における配線領域の割合が非常に大きくなる。また、接続部にパス・トランジスタを使用する、ため配線部分の遅延が非常に大きい。

そこで、上記ＦＰＧＡの欠点をカバーするプログラム可能な論理回路として、ＬＵＴカスケード（LUT cascade）が提案されている（引用文献［２］，［３］参照）。ＬＵＴカスケードとは、ＬＵＴを直列接続したものであり、演算処理を行う論理関数（目的論理関数）を関数分解して得られる複数の分解関数のＬＵＴを直列接続した形で表現できる。ＦＰＧＡは、論理回路をそのまま２次元的なＣＬＢのネットワークとして実現するのに対し、ＬＵＴカスケードは、目的論理関数を１次元的に直列接続したＬＵＴで実現する点において異なる。

つまり、ＦＰＧＡで用いられる各ＣＬＢは、基本的な論理ゲートを実現するものである。そして、これらの基本論理ゲートを配線により結合することで、複雑な論理回路を実現することに基本的な設計思想がある。従って、必然的に、ＦＰＧＡをチップ上にレイアウトする場合、２次元的に配置されたＣＬＢ間を自在に配線接続するための配線ネットワークを構成する必要がある。そのため、論理領域のほかに広い配線領域が必要となり、レイアウト面積が大きくなりがちである。

それに対し、ＬＵＴカスケードにおいては、一般に、多入力多出力の複雑な論理関数を表現するＬＵＴを、直列的（１次元的）に接続して、複雑な論理回路を実現することに基本的な設計思想がある。そのため、各ＬＵＴの出力と入力は、一般に複数の配線により接続される。そして、２次元的な配線ネットワークは不要であるため、一般に配線領域は少なく、チップ面積の殆どがＬＵＴを格納するためのメモリ領域となる。

また、配線のための、チップ面積や、配線部分の遅延時間もＦＰＧＡに比べて少ない。

以下、ＬＵＴカスケードの一設計法について簡単に説明する。第５２図はＬＵＴカスケードの原理を表す図である。なお、ここでは簡単のため、中間出力のないＬＵＴカスケードについて説明を行う。

ＬＵＴカスケードにおいて論理関数の演算を行う場合、まず、演算を行う論理関数（目的論理関数）f(X)をs個（s≧2）の分解関数$\{f_i(X_i)；i=0, \cdots, s-1\}$に関数分解する。ここで、$X=(x_0, \cdots, x_{n-1})$は入力変数を表す。また、入力変数の集合を{X}で表す。ここで、$\{X\}=\{X_0\} \cup \cdots \cup \{X_{s-1}\}$, $\{X_i\} \cap \{X_j\}=\phi$ $(i \neq j;\ i, j \in \{0, \cdots, s-1\})$である。各分解関数$f_i$の出力変数（一般に、ベクトル）を$Y_{i+1}$で表す。以下、X, Yの変数の個数を$|X|$, $|Y|$のように表すこととし、特に、$|X|=n$, $|X_i|=n_i$, $|Y_i|=u_i$と表記する。

任意の目的論理関数fは、$n_0=k$（kは分解関数f_0の入力変数の数）、$n_i=k-u_i$ $(0<i<s-1)$, $n_{s-1}=k-u_{s-1}-t$ $(0 \leqq t \leqq k-2)$となるように関数分解することが可能である（引用文献［２］，［３］参照）。そこで、目的論理関数fをs-1個のk入力の分解関数f_j $(j \in \{0, \cdots, s-2\})$と１個のk-t入力の分解関数$f_{s-1}$とに分解する。そして、各分解関数$f_r$ $(r \in \{0, \cdots, s-1\})$を分解表により表し、これをＬＵＴ（look up table）とする。ここで、目的論理関数$f_r(X_r)$の分解表（decomposition chart）とは、$2^{|Xr|}$列$2^{|fr|}$行の表で、各行、各列に２進符号のラベルをもち、その要素がf_rの対応する真理値であるような表をいう。

第５２図において、s個のＬＵＴ（LUT_0～LUT_{s-1}）は、それぞれ、分解関数$\{f_i；i=0, \cdots, s-1\}$を表現したものである。それぞれのＬＵＴ（LUT_r）は、k入力u_{r+1}出力のメモリ（以下、「論理関数メモリ」という。）を用いて実現することができる。このような論理関数メモリを、第５２図に示すようにカスケード接続することにより、Ｌ

ＵＴカスケードが実現される。

ＬＵＴカスケードでは、目的論理関数fの入出力変数の個数が大きい場合には、多数のＬＵＴをカスケード接続する必要がある。そのため、最適化されたＦＰＧＡに比べれば演算速度は遅くなる場合もある。しかし、ＬＵＴカスケードでは、各分解関数の演算は、論理関数メモリを用いて高速に行われる。そのため、ＭＰＵ上で動くプログラムに比べると演算速度を高速化することができる。

そして、ＬＵＴの段数によって、一義的に演算速度が定まる。そのため、論理回路の設計時に目的論理関数の演算時間を正確に予測することが容易である。また、ＬＵＴカスケードでは、互いに隣接する論理関数メモリ間でのみ配線を行えばよい。そのため、論理回路の設計時には配線遅延等の影響を考慮する必要がない。従って、ＬＵＴカスケードはＦＰＧＡに比べると論理回路設計が格別に容易になる。

しかしながら、実際に第５２図に示したようなＬＵＴカスケードにより目的論理関数の演算を行う場合には、目的論理関数を関数分解して得られる分解関数のＬＵＴ間のレイル数（二つのＬＵＴ間の線数（中間変数の数））や各ＬＵＴの入力変数の個数が目的論理関数により異なる。そのため、種々の目的論理関数を論理回路で実現しようとした場合、入力変数の個数及びレイル数が異なる種々のＬＵＴカスケードの論理回路が必要となり、実用性に欠ける。

そこで、多様な目的論理関数に対応可能なプログラマブル論理デバイスとするためには、各論理関数メモリの入力のビット数に余裕をもたせるべく、あらかじめビット数を十分大きくとっておくことが考えられる。しかし、論理関数メモリのメモリ容量は、論理関数メモリの入力が１増加するごとに２倍に増大する。従って、入力のビット数に余裕をもたせる場合、使用されないメモリ領域が増大しメモリ領域の無駄が多くなる。また、メモリ容量の増大に伴いメモリのレイアウト面積が増大する。そのため、回路の高集積化が困難となり、回路消費電力も大きくなる。

そこで、本発明の目的は、目的論理関数に応じて、各論理関数メモリ間の入力変数の入力線数とレイル数とを柔軟に変化させることが可能で、論理関数メモリの入力数、延いては論理関数メモリのメモリ容量を必要最小限に抑えて設計することが可能なプログラマブル論理デバイスを提供することにある。

〔引用文献〕

［１］　米国再発行特許Ｒｅ．３４，３６３号明細書

［２］　笹尾勤，松浦宗寛，井口幸洋，“多出力関数のカスケード実現と再構成可能ハードウェアによる実現”，電子情報通信学会FTS研究会，FTS2001-8，pp. 57-64，三重大学(2001-04).

［３］　T. Sasao, M. Matsuura, and Y. Iguchi, ″A cascade realization of multiple-output function for reconfigurable hardware,″ International Workshop on Logic and Synthesis (IWLS01), Lake Tahoe, CA, June 12-15, 2001. pp.225-230.

発明の開示

本発明に係るプログラマブル論理デバイスの第１の構成は、以下の構成を含むことを特徴とする：

（１）論理関数のＬＵＴ（look up table）を記憶するための、直列に順序づけて配列された論理関数メモリ；

（２）前記各論理関数メモリに対する入力変数が入力される複数の外部入力線；

（３）二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ；

（４）二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段

の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

各段の論理関数メモリには、目的論理関数をＬＵＴカスケードにより表現するために各ＬＵＴが表現する論理関数を格納しておく。なお、ここで「メモリ」とは、各アドレスに対応して一つのデータが記憶されており、アドレスを指定することによってそのアドレスに対応するデータを読み出す装置をいう。従って、「メモリ」は物理的に１個単体をなすメモリ素子に限らず、複数のメモリ素子を組み合わせて構成したものであってもかまわない。各接続回路は、接続メモリが出力する接続情報に従って、前段の論理関数メモリの出力線又は外部入力線の何れかと、後段の論理関数メモリの各入力線との接続関係を構成する。これにより、ＬＵＴカスケードが実現できる。

演算処理を行うＬＵＴカスケードに合わせて論理関数メモリ内の論理関数及び接続メモリ内の接続情報を書き換える。これにより、後段の論理関数メモリの入力線に接続するための前段の論理関数メモリの出力線の数、及び外部入力線の数を自在に変更できる。そして、ＬＵＴカスケードを実現する論理回路を自在に再構成することが可能となる。すなわち、目的論理関数に応じて、隣接する二つのＬＵＴ間を結ぶレイル数と外部からの入力変数の入力数との割合を任意に変更することが可能となる。そのため、１つのプログラマブル論理デバイスで実現可能な目的論理関数の範囲を広げることができる。その結果、各論理関数メモリの入力線の数を減らすことが可能となる。従って、回路を小型化することもできる。

また、各論理関数のＬＵＴの出力数と外部入力線からの入力変数の個数との割合を最適化することができる。そのため、論理関数メモリの入力線の配分、メモリの使用領域の無駄をなくすことができる。

なお、本発明に係るプログラマブル論理デバイスは、論理関数メモリが直列（一次元的）に配列されている点において、ＦＰＧＡとは決定的に異なる。

すなわち、ＦＰＧＡの場合、ファン・アウト（fan-out）があるので、ＣＬＢ間の入出力線の接続関係は１対１対応とはならない。従って、あらゆる接続関係に対応できるように接続回路の自由度を大きくする必要がある。そのため、接続回路が大規模となり、チップ内に占める配線領域の割合が非常に大きい。

それに対し、本発明に係るプログラマブル論理デバイスは、論理関数メモリが直列配列されている。従って、接続回路は、１つの論理関数メモリの出力線及び外部入力線と、１つの論理関数メモリの入力線とを選択的に対応づけて接続すればよい。

すなわち、前段の論理関数メモリの出力線の集合を$Y_1=\{y_1\}$、外部入力線の集合を$X=\{x\}$、後段の論理関数メモリの入力線の集合を$Z_{r+1}=\{z_{r+1}\}$とする。このとき、接続回路は、$Z_{r+1}\subseteq Y_1 \cup X$となるように、集合$Y_1 \cup X$の要素のうちの$|Z_{r+1}|$個が集合$Z_{r+1}$の各要素に一対一対応するように接続を行う。すなわち、接続回路は、基本的には$|Y_1 \cup X|$個の線から$|Z_{r+1}|$個の線を選択する機能（また、必要に応じて、選択した$|Z_{r+1}|$個の線の順序を変更する機能）を有するものであればよい。

従って、ハードウェア的には、接続回路は、ＦＰＧＡの接続回路に比べ、極めて簡単な構成となる。そのため、チップ内に占める配線領域も極めて小さくすることができる。また、２つの論理関数メモリ間のみの接続であり、配置配線設計は不要である。従って、回路設計において物理的な配線遅延の影響を考慮する必要もない。

更に、チップ内における配線領域が限定的である。そのため、高いクロック周波数で使用する場合にも、予め反射やクロストーク等の伝送路の周波数特性や寄生効果の影響を受けにくく設計しておくことが可能である。すなわち、高速回路としても使用することが可能となる。これは、ＦＰＧＡにより複雑な論理回路を設計した場合に、高いクロック周波数で使用できる回路の実現が困難になることとは対照的である。

本発明に係るプログラマブル論理デバイスの第２の構成は、以下の構成を含むことを特徴とする：

（１）論理関数のＬＵＴを記憶するための、リング状に配列された論理関数メモリ；

（２）前記各論理関数メモリに対する入力変数が入力される複数の外部入力線；

（３）二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ；

（４）二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

このように、論理関数メモリをリング状に接続することによって、後述（実施例１１〔例３〕参照。）のように、入力変数の一部又は全部が共通する複数の組み合わせ論理関数を、ＬＵＴリングとして１つのプログラマブル論理デバイスで同時に実現させることが可能となる。

また、後で述べるように、論理関数メモリのアドレスを保持するラッチを付加し、クロック・パルスと同期させ、論理関数メモリをリング状に接続することにより、任意段数のＬＵＴカスケードを模擬することが可能となる。これにより、論理関数メモリの使用効率をより向上させることができる。

なお、本発明に係るプログラマブル論理デバイスは、ＦＰＧＡと異なり、論理関数メモリは一次元的なリング状に配列されている。従って、接続回路は、１つの論理関数メモリの出力線及び外部入力線と、１つの論理関数メモリの入力線とを選択的に対応づけて接続すればよい。従って、第１の構成の場合と同様、ＦＰＧＡの接続回路に比べて接続回路は極めて簡単となり、配線領域も小さくすることができる。また、２つの論理関数メモリ間のみの接続であるため、配置配線設計は不要である。従って、論理設計において物理的な配線遅延の影響を考慮する必要はない。

本発明に係るプログラマブル論理デバイスの第３の構成は、前記第１又は２の構成

において、論理関数の演算結果を外部回路に出力するための外部出力線を備え；

前記接続メモリは、二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線のうち前記外部出力線に接続されるものを選択するための接続情報をも記憶するものであり；

前記接続回路は、二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線と前記外部出力線との接続をも行うものである；

ことを特徴とする。

この構成により、直列接続された論理関数メモリうちの任意のものから、その出力を取り出すことができる。従って、ＬＵＴカスケードの段数が少ない場合には、論理関数メモリ列の途中の出力を出力変数として取り出すことで、演算処理を早期に終了させて演算速度を高速化することができる。

また、多出力論理関数の演算を行う場合には、ＬＵＴカスケードの途中で確定した演算結果を、後段のＬＵＴに入力することなく、途中で出力させることができる。これにより、後段のＬＵＴの入力数を削減することができる。また、演算結果が確定した時点でそれを出力することで、演算処理速度を上げることができる。更に、後段のＬＵＴの入力数が減ることで、後段のＬＵＴに必要とされるメモリ容量の削減が可能となる。

本発明に係るプログラマブル論理デバイスの第４の構成は、前記第１乃至３の何れか一の構成において、論理関数メモリのメモリ領域を指定する領域指定変数を記憶する領域指定記憶手段を備え；

前記接続回路は、二つの前記論理関数メモリ間において、領域指定記憶手段の出力に従って、

前段の前記論理関数メモリの出力及び外部入力線からの入力変数が、領域指定変数

により特定される後段の前記論理関数メモリのメモリ領域に入力されるように、

又は、領域指定変数により特定される前段の前記論理関数メモリのメモリ領域からの出力及び外部入力線からの入力変数が、後段の前記論理関数メモリに入力されるように、

前段の前記論理関数メモリの出力線、外部入力線、及び領域指定記憶手段の出力線と、後段の前記論理関数メモリの入力線との接続を行うものである；

ことを特徴とする。

この構成によれば、各論理関数メモリにＬＵＴを格納する場合に、メモリを複数の領域（ページ等）に区切って複数のＬＵＴを記憶させておく。領域指定記憶手段には、それぞれの論理関数のＬＵＴが格納されたメモリ領域を指定する領域指定変数の値を記憶させておく。接続メモリには、それぞれの論理関数に対応する領域指定変数が出力される領域指定記憶手段の出力線が、後段の論理関数メモリの入力線と接続するように接続情報を記憶させておく。

そして、所望の論理関数のＬＵＴを読み出す場合には、その論理関数に対応する領域指定変数、領域指定記憶手段から、接続情報を接続メモリから読み出し、接続回路の接続関係を切り替える。それとともに、各論理関数メモリの領域指定変数で指定されるメモリ領域のＬＵＴを読み出す。これにより、複数の目的論理関数の演算を切り替えて行うことが可能となる。

本発明に係るプログラマブル論理デバイスの第５の構成は、前記第１乃至４の何れか一の構成において、前記論理関数メモリの入力側又は出力側に設けられ、外部から入力されるデータ・ストローブ信号に従って、前記論理関数メモリへの入力又は前記論理関数メモリの出力を取り込んで一時的に保持する中間変数レジスタ；

を備えていることを特徴とする。

これにより、各段の論理関数メモリの出力は中間変数レジスタで一時的に保持され

る。そのため、演算処理の流れは中間変数レジスタで一時的に止められる。すなわち、データ・ストローブ信号に同期して、各論理関数メモリを１段ずつＬＵＴによる演算が行われる。従って、データ・ストローブ信号を外部回路のクロックなどに同期させれば、各論理関数のＬＵＴの演算を外部回路に同期させることができる。

また、論理関数メモリごとに演算が１段ずつ行われる。そのため、複数のデータ（ジョブ）に対する演算をパイプライン処理で実行させることが可能となる。

また、論理関数メモリがリング状に接続されている場合、演算可能な論理関数の入力数に関する自由度を大きくすることができる。すなわち、目的論理関数の入力変数の個数が各論理関数メモリの入力線数の総和よりも多い場合、クロック・パルスを使用することにより、論理関数メモリのリングを一周以上用いてＬＵＴカスケードを構成する。そして、同じ論理関数メモリの使用領域を必要に応じて変化させて関数演算を順次行わせればよい。従って、目的論理関数やメモリ量の数に応じてＬＵＴカスケードの段数を柔軟に変化させることが可能となる。そのため、設計自由度が大きくなる。

更に、論理関数メモリがリング状に接続されている場合、クロック・パルスを使用することにより、ＬＵＴカスケードの段数を多くすることができる。そのため、各段のＬＵＴを小さくすることができる。従って、各段の論理関数メモリは、メモリ容量の比較的小さいものを使用することができる。そのため、各論理関数メモリの消費電力は低く抑えることが可能である。更に、演算の流れに従って、一部の論理関数メモリのみがデータ読み出しの動作をする。そして、この動作を行うメモリのみが主として電力を消費する。従って、ＭＰＵやＦＰＧＡに比べ、低消費電力で動作させることが可能となる。また、データ読み出しの動作をしていないメモリは、低消費電力状態（スリープ状態）にするように構成すれば、更に消費電力の低減が可能となる。

ここで、データ・ストローブ信号としてクロック・パルスを使用することができる

。

本発明に係るプログラマブル論理デバイスの第６の構成は、前記第１乃至５の何れか一の構成において、前記中間変数レジスタと並列に接続されたバイパス線；

及び、前記中間変数レジスタの出力側に設けられ、前記中間変数レジスタの出力線又は前記バイパス線の何れか一方を選択し、選択された線の信号を出力するバイパス選択回路；

を備えていることを特徴とする。

この構成により、各バイパス選択回路で、中間変数レジスタの出力線を選択すれば、データ・ストローブ信号を使用した同期的な演算処理を行わせることができる。また、各バイパス選択回路で、バイパス線を選択すれば、前段の論理関数メモリの出力が、中間変数レジスタで止められることなく直接後段の論理関数メモリに送られる。従って、ＬＵＴカスケードによる目的論理関数の演算を非同期的に高速で実行することができる。

本発明に係るプログラマブル論理デバイスの第７の構成は、前記第５又は６の構成において、前記データ・ストローブ信号を計数し、演算を実行する前記論理関数メモリの番号を特定する論理関数メモリ特定手段；

を備えていることを特徴とする。

このように、論理関数メモリ特定手段を設けることで、演算の実行中において、ジョブが存在する論理関数メモリの位置を特定することができる。ここで、「ジョブ」とは、入力変数に対してＬＵＴカスケードの各ＬＵＴにより演算を実行しデータを生成する作業をいう。

本発明に係るプログラマブル論理デバイスの第８の構成は、前記第５乃至７の何れか一の構成において、演算処理を実行させる前記論理関数メモリを通常動作状態とし、それ以外の前記論理関数メモリを低消費電力状態とする制御を行う電源制御手段；

を備えていることを特徴とする。

この構成によれば、各段の論理関数メモリによる演算処理は中間変数レジスタで一時的に止められながら、データ・ストローブ信号に同期して１段ずつ実行される。この際、電源制御手段は、演算処理に使用されていない論理関数メモリは低消費電力状態（スリープ状態）とし、演算処理に使用する論理関数メモリのみを通常動作状態（ウェイク・アップ状態）とする。これにより、プログラマブル論理デバイスの消費電力を低減させることもできる。

本発明に係るプログラマブル論理デバイスの第９の構成は、前記第１乃至８の何れか一の構成において、前記各論理関数メモリの一部の入力線は、前記接続回路を介すことなく前記外部入力線に直接接続されていることを特徴とする。

最初の段の論理関数メモリについては、総ての入力に対して入力変数が入力される。一方、最初の段以外の各論理関数メモリにおいては、その入力の一部には前段の論理関数メモリの出力が入力され、他の入力には、入力変数が入力される。

ところで、実用的なＬＵＴカスケードの多くにおいて、最初の段以外の各論理関数メモリには、最低限１つの入力変数が入力される。そこで、プログラマブル論理デバイスを汎用目的で製作する場合には、最初の段以外の各論理関数メモリの入力のうちの少なくとも１つに、接続回路を介すことなく入力変数を直接入力するように構成する。これにより、接続回路の入力線の数を減らすことができる。また、接続回路及び接続メモリの出力の配線数を減らすことができるため、回路を更に小型化することが可能となる。

本発明に係るプログラマブル論理デバイスの第１０の構成は、前記第１乃至９の何れか一の構成において、前記各論理関数メモリの一部の出力線は、前記接続回路を介すことなくその後段の論理関数メモリの一部の入力線に直接接続されていることを特徴とする。

最初の段以外の各論理関数メモリにおいては、その入力の一部には前段の論理関数メモリの出力が入力され、他の入力には、入力変数が入力される。

ところで、実用的なＬＵＴカスケードの多くにおいて、最初の段以外の各論理関数メモリには、最低１つは前段の論理関数メモリの出力が入力される。そこで、プログラマブル論理デバイスを汎用目的で製作する場合には、最初の段以外の各論理関数メモリの入力のうちの一部を、接続回路を介すことなく前段の論理関数メモリの出力を直接入力するように構成する。これにより、接続回路の入力線の数を減らすことができる。また、接続回路及び接続メモリの出力の配線数を減らすことができるため、回路を更に小型化することが可能となる。

本発明に係るプログラマブル論理デバイスの第１１の構成は、前記第１乃至１０の何れか一の構成において、前記接続回路は、複数のセレクタ回路を含み；

前記各セレクタ回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線と前記外部入力線とのうち何れか一つ、又は、前段の前記論理関数メモリの出力線と前記外部入力線と領域指定記憶手段の出力線とのうち何れか一つを選択して後段の前記論理関数メモリの入力線に接続するものであること；

を特徴とする。

本発明に係るプログラマブル論理デバイスの第１２の構成は、前記第１乃至１１の何れか一の構成において、前記接続回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線の接続順序をシフトして後段の前記論理関数メモリの入力線に接続するシフタ回路を含むことを特徴とする。

本発明に係るプログラマブル論理デバイスの第１３の構成は、前記第１乃至１２の何れか一の構成において、前記接続回路は、複数のマルチプレクサを含み；

前記マルチプレクサは、前記接続メモリの出力値に従って、前段の前記論理関数メモリの複数の出力線及び複数の前記外部入力線のうちの何れか一本を選択して後段の

前記論理関数メモリの入力線に接続するものであること；

を特徴とする。

図面の簡単な説明

第１図は本発明の実施例１に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第２図は第１図の入力変数選択回路の構成を表す回路図である。

第３図は第１図の論理関数メモリの構成を表すブロック図である。

第４図は第１図の接続回路の周辺の回路ブロック図である。

第５図は第１図の接続メモリの構成を表すブロック図である。

第６図は実施例１に係るプログラマブル論理デバイスの動作を表すフローチャートである。

第７図は第１図の接続回路にクロスバ・スイッチを使用した場合の回路ブロック図である。

第８図は第１図の接続回路にマルチプレクサ・アレイを使用した場合の回路ブロック図である。

第９図は第１図の接続回路にセレクタ・アレイを使用した場合の回路ブロック図である。

第１０図は８ビット加算器の論理関数を表す図である。

第１１図は８つの分解関数に関数分解した８ビット加算器を表す図である。

第１２図は８つの分解関数に関数分解した８ビット加算器を表す図である。

第１３図は４つの分解関数に関数分解した８ビット加算器を表す図である。

第１４図は４つの分解関数に関数分解した８ビット加算器の各分解関数の真理値である。

第１５図は例１において論理関数メモリ論理関数メモリ４－０の０ページ目に記憶された真理値である。

第１６図は例１において論理関数メモリ論理関数メモリ４－１～４－３の０ページ目に記憶された真理値である。

第１７図は本発明の実施例２に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第１８図は本発明の実施例３に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第１９図は本発明の実施例４に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第２０図は第１６図の出力レジスタ及び出力デコーダの構成を表すブロック図である。

第２１図は第１７図の記憶素子の構成を表すブロック図である。

第２２図は本発明の実施例５に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第２３図は第１９図の第２の出力選択回路の構成を表すブロック図である。

第２４図は本発明の実施例６に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第２５図は本発明の実施例７に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第２６図は第２５図の演算制御部１０の構成を表すブロック図である。

第２７図は第２６図の出力コントローラ６４の内部構成を表すブロック図である。

第２８図は実施例７に係るプログラマブル論理デバイスの動作を表すフローチャートである。

第２９図は実施例７に係るプログラマブル論理デバイスの動作時における各信号の変化を表すタイミング図である。

第３０図は２ｎビットの二進数A, Bの加算を行う論理関数fを表す図である。

第３１図は２ｎビットの二進数A, Bの加算を行う論理関数fを２ｎ個の分解関数{g_0, g_1, …, g_{2n-1}}に関数分解した図である。

第３２図は２ｎビットの二進数A, Bの加算を行う論理関数fを２ｎ個の分解関数{g_0, g_1, …, g_{2n-1}}に関数分解した図である。

第３３図は２ｎビットの二進数A, Bの加算を行う論理関数fをｎ個の分解関数{ f_0, f_1, …, f_{n-1}}に関数分解した図である。

第３４図は分解関数f_iの真理値表である。

第３５図は論理関数メモリ４－０の第１ページに記憶する真理値表である。

第３６図は論理関数メモリ４－０～４－３の第０ページに記憶する真理値表である。

第３７図は本発明の実施例８に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第３８図は本発明の実施例９に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第３９図は本発明の実施例１０に係るプログラマブル論理デバイスの全体構成を表す図である。

第４０図は本発明の実施例１０に係るプログラマブル論理デバイスの演算部の構成を表すブロック図である。

第４１図は第４０図の接続回路とメモリ・アドレス・レジスタの構成を表す図である。

第４２図は本発明の実施例１０に係るプログラマブル論理デバイスの出力回路の構成を表す図である。

第４３図は実施例１０に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。

第４４図は実施例１０に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。

第４５図は実施例１０に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。

第４６図はメモリ・パッキングの概念を説明する図である。

第４７図は本発明の実施例１１に係るプログラマブル論理デバイスの構成を表す図である。

第４８図は２個の論理回路メモリを用いたプログラマブル論理デバイスで４段のＬＵＴカスケードを実現する例を表す図である。

第４９図は論理関数ｆ，ｇをＬＵＴカスケードで表現した図である。

第５０図は論理関数ｆ，ｇをＬＵＴカスケードで表現した図である。

第５１図は論理関数ｆ，ｇを合成したＬＵＴリングを表す図である。

第５２図はＬＵＴカスケードの原理を表す図である。

発明を実施するための最良の形態

〔用語の定義〕

最初に、以下の説明で使用する記号及び用語について定義しておく。

（１）記号「{　}」は非順序集合（unordered set）を表す。記号「（　）」は順序集合（ordered set）を表す。変数の集合$\{x_1, x_2, \cdots, x_n\}$について、集合の各要素の順序を考慮する場合には、$(x_1, x_2, \cdots, x_n)=X$と記す。順序を考慮しない場合には、$\{x_1, x_2, \cdots, x_n\}=\{X\}$と記す。Xの変数の個数を$|X|$で表す。

（２）プログラマブル論理デバイスによりＬＵＴカスケードを実現しようとする論理

関数のことを「目的論理関数」といい、fと記す。目的論理関数の入力変数の集合を$X=(x_1, x_2, \cdots, x_n)$（但し、nは自然数）と記す。

（３）目的論理関数f(X)が、$f(X)=f_{s-1}(X_{s-1}, f_{s-2}(X_{s-2}, f_{s-3}(\cdots f_1(X_1, f_0(X_0))\cdots)))$，$\{X_i\} \subseteq \{X\}$，$\{X\}=\{X_0\} \cup \{X_1\} \cup \cdots \cup \{X_{s-1}\}$のような合成論理関数で表されるとする。このとき、目的論理関数f(X)を順序づけられた複数の関数の集合$(f_0, f_1, \cdots, f_{s-1})$に分解することを「関数分解（function decomposition）」という。目的論理関数f(X)を関数分解して得られる各関数$\{f_i ; i \in \{0, 1, 2, \cdots, s-1\}\}$を「分解関数（decomposition function）」という。

（４）「多出力論理関数」とは、出力変数の個数が複数である論理関数をいう。

以下、本発明を実施するための最良の形態について、図面を参照しながら説明する。

（実施例１）

第１図は本発明の実施例１に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本発明の実施例１に係るプログラマブル論理デバイスは、入力変数レジスタ１、入力変数選択回路２－０～２－３、入力選択メモリ３－０～３－３、論理関数メモリ４－０～４－３、接続回路５－１～５－３、接続メモリ６－１～６－３、及び演算制御部１０を備えている。

入力変数レジスタ１は、目的論理関数f(X)の演算に用いられるn個の入力変数$X=(x_1, \cdots, x_n)$（但し、nは自然数）を記憶する。尚、入力変数Xは、外部入力線より入力変数レジスタ１に入力される。論理関数メモリ４－０～４－３は、目的論理関数fを関数分解して得られる分解関数$\{f_i ; i \in \{0, 1, 2, 3\}\}$の真理値表を、ＬＵＴとして記憶する。尚、本実施例においては、説明の便宜上、論理関数メモリ４－ｉ（$i \in \{0, 1, 2, 3\}$）の段数を４段としているが、一般にはこの段数は任意に設定することができる。これらの論理関

数メモリ４－０～４－３は、直列に配列され、それぞれ接続回路５－１～５－３を介して直列に順序づけて配列されている。

入力変数選択回路２－０～２－３は、入力変数レジスタ１から出力されるn個の入力変数$X=(x_1, \cdots, x_n)$のうちから、それぞれ、各段の論理関数メモリ４－０～４－３に記憶された分解関数$\{f_i ; i \in \{0, 1, 2, 3\}\}$の真理値表の入力変数$X_i$（$i \in \{0, 1, 2, 3\}$）を選択し、論理関数メモリ４－０又は接続回路５－１～５－３へ出力する。入力選択メモリ３－０～３－３は、それぞれ、入力変数選択回路２－０～２－３の入力変数の選択に関する情報（以下、「入力選択情報」という。）がＬＵＴとして記憶されている。入力変数選択回路２－０～２－３は、各入力選択メモリ３－０～３－３から出力される入力選択信号に基づいて、入力変数の選択の切り換えを行う。

各接続回路５－ｉ（ｉ∈｛１，２，３｝）は、論理関数メモリ４－（ｉ－1）及び入力変数選択回路２－ｉより入力される、中間変数Y_i及び入力変数X_iを、選択するとともに適宜順序づけて、後段の論理関数メモリ４－ｉ及び外部出力線７－ｉへ接続する。各接続メモリ６－ｉは、各接続回路５－ｉの接続関係に関する情報（以下、「接続情報」という。）を記憶している。接続回路５－ｉは、接続メモリ６－ｉから出力される接続情報信号に基づき、接続関係の再構成を行う。

演算制御部１０は、このプログラマブル論理デバイス全体の演算処理の制御を行う。

第２図は第１図の入力変数選択回路２－ｉ（$i \in \{0, 1, 2, 3\}$。以下、符号２－０～２－３をまとめていう場合には、符号２と記す。）の構成を表す回路図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

本実施例の入力変数選択回路２－ｉ（$i \in \{0, 1, 2, 3\}$）は、第２図のようなシフタ回路により構成されている。尚、第２図では、説明の便宜上、１７入力８出力の入力変数選択回路２－ｉの例を示しているが、入出力の数は、これに限られるものではない。

入力変数選択回路２－ｉ（ｉ∈{0, 1, 2, 3}）の入力端子in(00)～in(16)は、入力変数レジスタ１の出力端子に接続されている。従って、入力端子in(00)～in(16)からは入力変数$X=(x_1, \cdots, x_n)$が入力される。また、入力変数選択回路２－０の出力端子out(00)～out(07)は、論理関数メモリ４－０の入力に接続されている。入力変数選択回路２－ｉ（ｉ∈{1, 2, 3}）の出力端子out(00)～out(07)は、接続回路５－ｉの入力側端子の一部に接続されている。

入力変数選択回路２－ｉ（ｉ∈{0, 1, 2, 3}）は、入力側から、８ビットシフト回路１１－３、４ビットシフト回路１１－２、２ビットシフト回路１１－１、及び１ビットシフト回路１１－０が直列に接続されている。これにより、入力変数選択回路２－ｉ（ｉ∈{0, 1, 2, 3}）は、入力端子in(00)～in(16)の内容を0～15ビットの範囲でシフトさせて、連続する８ビット分を出力端子out(00)～out(07)に出力させることができる。

各シフト回路１１－０～１１－３には、１ビットの制御線shf0～shf3が接続されている。制御線shf ｊ（ｊ∈{0, 1, 2, 3}）が0のときは、シフト回路１１－ｉは接続のシフトを行わず、制御線shf ｊ（ｊ∈{0, 1, 2, 3}）が1のときは、シフト回路１１－ｉは接続のシフトを行う。尚、入力変数選択回路２－ｉ（ｉ∈{0, 1, 2, 3}）の各制御線shf0～shf3は、入力選択メモリ３－ｉの出力に接続されている。すなわち、入力選択メモリ３－０から読み出された４ビットのメモリの内容（すなわち、入力選択情報）が、選択制御信号としてそのまま制御線shf0～shf3に出力され、入力変数選択回路２－ｉ（ｉ∈{0, 1, 2, 3}）における入力変数のシフト量が設定される。

尚、第２図では、簡略化のために省略しているが、シフタ回路１１－０～１１－３内の各パス・トランジスタにより入力変数Xの信号振幅が減衰する。従って、実際には、入力変数Xの信号振幅を補償するために、シフタ回路の数段ごとに増幅器（バッファ）を挿入しておくことが必要となる。

また、多出力論理関数を、複数の出力集合に分割して実現する場合には、入力変数

選択回路２－ｉ（ｉ∈{0, 1, 2, 3}）にシフタが使用できない場合がある。このような場合、接続回路を第７図に示すクロスバ・スイッチや第８図に示すマルチプレクサ・アレイ（後述）と同様のものを使用してもよい。

第３図は第１図の論理関数メモリ４－ｉ（ｉ∈{0, 1, 2, 3}．以下、符号４－０～４－３をまとめていう場合には、符号４と記す。）の構成を表すブロック図である。

論理関数メモリ４は、その内部に、ｐページからなるメモリ領域FM_0～FM_{p-1}を備えている。これらのメモリ領域には、分解関数$f_i^{(j)}$（$i \in \{0, 1, 2, 3\}$，$j \in \{0, \cdots, p-1\}$）の真理値表がＬＵＴ（図では「$LUT^{(j)}$」と記す。）として格納されている。以下では、$LUT^{(j)}$が格納されているメモリ領域をFM_jと記す。

尚、ここで、添字「$^{(j)}$」（$j \in \{0, \cdots, p-1\}$）は、ｊ番目の目的論理関数$f^{(j)}$を表す。本実施例におけるプログラマブル論理デバイスでは、複数の目的論理関数$f^{(j)}$の分解関数$\{f_i^{(j)}\}$を、それぞれの論理関数メモリ４の第ｊページに記憶させておく。そして、使用目的に応じて、ページを切り換えて、目的論理関数を選択することが可能な構成とされている。

また、論理関数メモリ４－ｉ（ｉ∈{0, 1, 2, 3}）は、アドレス・デコーダ１６を備えている。アドレス・デコーダ１６は、演算制御部１０から入力されるページ選択番号p_rに基づいて、メモリ領域FM_rのメモリ・アクセスを可能とする。また、アドレス・デコーダ１６は、接続回路２からN_{in}ビットの入力線１７を介して入力される変数(X_r, Y_r)（$r \in \{0, 1, 2, 3\}$．但し、$r=0$のとき、$Y_r=\phi$。）に基づき、メモリ領域FM_r内のメモリ内容を読み出すアドレスを選択する。この選択されたアドレスのメモリ・セルには、分解関数$f_r^{(j)}$の真理値$f_r^{(j)}(X_r, Y_r)$が格納されている。メモリ領域FM_rは、アドレス・デコーダ１６により、メモリ・セル内のアドレスが指定されると、当該メモリ・セル内に記憶されているデータを、中間変数$Y_{r+1}=(y_{r+1,1}, \cdots, y_{r+1,|Y_{r+1}|})$として出力線１８に出力する。

第４図は第１図の接続回路５－ｉ（ｉ∈{1, 2, 3}．以下、符号５－１～５－３をまとめていう場合には、符号５と記す。）の周辺の回路ブロック図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

本実施例における接続回路５は、入力線i_0～i_{15}からの入力信号を循環的に任意のビット数だけシフトさせて出力線o_1～o_{15}に出力するシフタ回路により構成されている。尚、第４図においては、説明の便宜上、一例として、１６ビット入力１６ビット出力の接続回路を記載しているが、入出力のビット数は、特にこれに限定されない。

接続回路５は、入力線i_0～i_{15}の側から、８ビットシフタ回路２０－３、４ビットシフタ回路２０－２、２ビットシフタ回路２０－１、１ビットシフタ回路２０－０が直列に接続された構成とされている。各シフタ回路２０－ｉ（ｉ∈{0, 1, 2, 3}）のオン／オフは、接続メモリ６（符号６－１～６－３をまとめて符号６と記す。以下同じ。）の出力に接続された制御線s_j（ｊ∈{0, 1, 2, 3}）により制御される。

接続回路５－ｒ（ｒ∈{1, 2, 3}）の入力線i_0～i_{15}のうち、i_0～i_7は前段の論理関数メモリ４－（ｒ－１）の出力側に接続されており、i_8～i_{15}は入力変数選択回路２－ｒの出力側に接続されている。また、接続回路５－ｒの出力線o_0～o_{15}のうち、o_0～o_7は、後段の論理関数メモリ４－ｒの入力側に接続されており、o_8～o_{15}は外部出力線７－ｒとなっている。

各シフタ回路２０－０～２０－３の接続関係のシフトがないときには、入力線i_0～i_{15}は、それぞれ出力線o_0～o_{15}に接続される。これにより、前段の論理関数メモリ４－（ｒ－１）の出力である中間変数Y_rは、総て後段の論理関数メモリ４－ｒに入力される。

一方、シフタ回路２０－３のみが接続関係を８ビットシフトさせた場合には、入力線i_0～i_7が、それぞれ、出力線o_8～o_{15}に接続され、入力線i_8～i_{15}が、それぞれ、出力線o_0～o_7に接続される。これにより、前段の論理関数メモリ４－（ｒ－１）の出力である中間変数Y_rは、総て外部出力線７－ｒに出力される。また、入力変数選択回路２－ｒから出力される入力変数X_rは、総て後段の論理関数メモリ４－ｒに入力される。

また、各シフタ回路２０-０～２０-２により、接続関係をjビット（１≦j≦７）だけシフトさせた場合には、入力線i_{15-j+1}～i_{15}，i_0～i_{j-1}が、それぞれ、出力線o_8～o_{15}に接続され、入力線i_j～i_{7+j}が、それぞれ、出力線o_0～o_7に接続される。これにより、入力線i_8～i_{7+j}に入力されるjビットの入力変数X_rが、後段の論理関数メモリ４-rに入力されるとともに、入力線i_j～i_7に入力される８-jビットの中間変数Y_rが後段の論理関数メモリ４-rに入力される。従って、後段の論理関数メモリ４-rに入力する入力変数X_rの変数の個数$|X_r|=n_r$と、後段の論理関数メモリ４-rに入力する中間変数Y_rの変数の個数$|Y_r|=u_r$とを、$k=n_r+u_r=8$の条件下で自由に変更することが可能となっている。

第５図は第１図の接続メモリ６の構成を表すブロック図である。接続メモリ６-ｉ（ｉ∈｛１，２，３｝）の内部には、ｐ個のメモリ領域CM_0～CM_{p-1}が備えられている。各メモリ領域CM_0～CM_{p-1}には、接続回路５-ｉの各シフタ回路２０-（N_c-1）～２０-０（ここで、N_cは接続回路におけるシフタ回路の段数を表し、第４図では$N_c=4$である。）のシフト情報を表す接続情報$(s_{Nc-1}{}^{(j)},\cdots,s_0{}^{(j)})$（ｊ∈｛０，…，p-1｝）が記憶されている。尚、ここで、添字「$^{(j)}$」（j∈{0,…,p-1}）は、ｊ番目の目的論理関数$f^{(j)}$に対応した接続情報であることを表す。

これらのメモリ領域CM_0～CM_{p-1}のN_cビットの出力線s_0～s_{Nc-1}は、接続回路５-ｉの各シフタ回路２０-０～２０-（N_c-１）の制御線となっている。

また、各接続メモリ６の内部には、アドレス・デコーダ２１が設けられている。アドレス・デコーダ２１は、演算制御部１０から入力されるページ選択番号p_rに従って、r番目のメモリ領域CM_rを選択する。アドレス・デコーダ２１により選択されたメモリ領域CM_rは、N_cビットの接続情報$(s_{Nc-1}{}^{(r)},\cdots,s_0{}^{(r)})$からなる接続制御信号を接続回路５に出力する。

以上のように構成された本実施例に係るプログラマブル論理デバイスにおいて、以下その動作について説明する。

第６図は実施例１に係るプログラマブル論理デバイスの動作を表すフローチャートである。

最初に、演算を行おうとする目的論理関数f(X)を関数分解した分解関数$\{f_0, \cdots, f_{s-1}\}$（$2 \leqq s \leqq 4$）を、各論理関数メモリ４－０～４－３の第rページに書き込んでおく。尚、ここで、rはページ番号を示す。また、各入力選択メモリ３－０～３－３及び各接続メモリ６－１～６－３の第rページには、上記各分解関数$\{f_0, f_1, \cdots, f_{s-1}\}$に対応した入力選択情報及び接続情報を書き込んでおく。尚、各メモリへの書き込み機能については、第１図では図示していないが、通常のメモリの書き込み方法により行われる。以下では、s=4であるものとして説明する。

以上の書き込みがされた状態において、まず、演算制御部１０は、各入力選択メモリ３－０～３－３、及び接続メモリ６－１～６－３に対して、ページ選択番号p_rを設定する（Ｓ１）。

これにより、各入力選択メモリ３－０～３－３は、第rページに書き込まれた入力選択情報を入力変数選択回路２－０～２－３の制御線shf0～shf3に出力する（Ｓ２）。また、接続メモリ６－１～６－３は、第rページに書き込まれた接続情報$(s_3^{(r)}, \cdots, s_0^{(r)})$を接続回路５－１～５－３の各制御線に出力する（Ｓ３）。

入力変数選択回路２－０～２－３は、制御線shf0～shf3に入力された入力選択情報に従って、入力端子in(00)～in(16)の一部を出力端子out(00)～out(07)と電気的に接続する。接続回路５－１～５－３は、各制御線より入力される接続情報$(s_3^{(r)}, \cdots, s_0^{(r)})$に従って、入力線$i_8$～$i_{15}$と出力線$o_0$～$o_7$とを接続する。

次に、演算制御部１０は、入力変数レジスタ１により、入力変数Xの出力を行う（Ｓ４）。これにより、論理関数メモリ４－０及び接続回路５－１～５－３には、それぞれ、入力変数X_0，X_1，X_2，X_3が出力される。

これに伴い、まず論理関数メモリ４－０は、入力変数X_0に対する分解関数f_0の真理値

$f_0(X_0)$を中間変数Y_1として出力する。この中間変数Y_1は、接続回路５－１を介して論理関数メモリ４－１の入力に伝達される。また、場合によっては、中間変数Y_1の一部は、外部出力線７－１へと伝達される。

論理関数メモリ４－１には、論理関数メモリ４－０から伝達された中間変数Y_1と、入力変数選択回路２－１及び接続回路５－１を介して入力変数レジスタ１から伝達される入力変数X_1が入力される。論理関数メモリ４－１は、入力変数X_1及び中間変数Y_1に基づき、分解関数f_1の真理値$f_1(X_1, Y_1)$を中間変数Y_2として出力する。この中間変数Y_2は、接続回路５－２を介して論理関数メモリ４－２の入力に伝達される。また、場合によっては、中間変数Y_2の一部は、外部出力線７－２へと伝達される。

論理関数メモリ４－２には、論理関数メモリ４－１から伝達された中間変数Y_2と、入力変数選択回路２－２及び接続回路５－２を介して入力変数レジスタ１から伝達される入力変数X_2が入力される。論理関数メモリ４－２は、入力変数X_2及び中間変数Y_2に基づき、分解関数f_2の真理値$f_2(X_2, Y_2)$を中間変数Y_3として出力する。この中間変数Y_3は、接続回路５－３を介して論理関数メモリ４－３の入力に伝達される。また、場合によっては、中間変数Y_3の一部は、外部出力線７－３へと伝達される。

論理関数メモリ４－３には、論理関数メモリ４－２から伝達された中間変数Y_3と、入力変数選択回路２－３及び接続回路５－３を介して入力変数レジスタ１から伝達される入力変数X_3が入力される。論理関数メモリ４－３は、入力変数X_3及び中間変数Y_3に基づき、分解関数f_3の真理値$f_3(X_3, Y_3)$を出力変数fとして出力する。この出力変数fは、外部出力線７－４に伝達される（Ｓ５）。以上の処理は、前段から後段に向かって直列に伝達しながらパイプライン処理により行われる。そして、目的論理関数ｆの演算結果は、各外部出力線７－１～７－４より取り出される。

なお、本実施例では接続回路５－１～５－３として、第４図のシフタ回路を使用したが、接続回路５－１～５－３としては、第７図に示したクロスバ・スイッチ（Cross

Bar Switch）による接続回路５－ｉ’（ｉ∈｛１，２，３｝）や第８図に示したマルチプレクサ・アレイ（Multiplexer Array）による接続回路５－ｉ”を使用することができる。要するに、前段の論理関数メモリの出力線の集合Y_{i-1}、外部入力線の集合X_i、後段の論理関数メモリの入力線の集合Z_iに対して、接続回路５－ｉは、集合$Y_{i-1} \cup X_i$の要素から$|Z_i|$個の要素を選択し、集合Z_iの要素と一対一に対応づける機能（又は、必要に応じて、選択した$|Z_i|$個の要素の順序を外部指示に従って置換し集合Z_iの要素と一対一に対応づける機能）を有するものであればよい。

尚、上記クロスバ・スイッチによる接続回路５－ｉ’は、シフタ回路に比べると切換制御線の数が多くなり回路が大きくなるという欠点はあるが、信号が通過するパス・トランジスタの段数を減らすことができるため、演算速度の高速化が可能となる。

また、接続回路５－１～５－３において、変数順序の置換を必要としない場合には、第９図に示したようなセレクタ・アレイ（Selector Array）による接続回路５－ｉ’’’を使用してもよい。セレクタ・アレイによる接続回路５－ｉ’’’は極めて簡単な回路で構成することが可能である。従って、接続回路５－ｉ’’’は、小さいレイアウト面積しか必要とせず、高速であり、回路消費電力も小さい。

また、本実施例では、各論理関数メモリ４－０～４－３を非同期に動作させる例を示したが、本発明においては、各論理関数メモリ４－０～４－３をクロックで同期させて動作させてもよい。

次に、上記動作をより分かりやすく説明するため、具体的な例を用いて本実施例のプログラマブル論理デバイスの具体的な動作の説明を行う。

〔例１〕加算回路

ここでは、簡単な例として、二つの８ビットの二進数$A=(a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$，$B=(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$の加算を行う加算器を、上記プログラマブル論理デバイスによって実行する例を説明する。尚、下位からの桁上げ（キャリ

一）入力ビットとしてc_{in}も考慮して、以下のような演算を行う加算器を考える。

（数１）

$$\begin{array}{cccccccccc} & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ & b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\ +) & & & & & & & & c_{in} \\ \hline c_{out} & S_7 & S_6 & S_5 & S_4 & S_3 & S_2 & S_1 & S_0 \end{array}$$

ここで、桁上げ入力ビットc_{in}とは、８ビット以上の大きな数の加算を行う場合に、加算器を直列につなげて使用する際に、下位の加算器からの桁上げを表すビットである。桁上げ出力ビットc_{out}も、同様に、上位の加算器に出力する桁上げを表すビットである。

このような二つの８ビットの二進数A, Bの加算を行う論理関数をf=f(X) (X=(A, B))とすると、論理関数fは第１０図により表される。この関数は、第１１図のように、８つの分解関数{g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7}に関数分解することが可能である。ここで、各分解関数は、（数２）又は（数３）のような論理式で表される。

（数２）

$$\begin{aligned} g_0 &= [c_{out}^{(0)}, S_0] \\ &= [a_0b_0 \vee a_0c_{in} \vee b_0c_{in},\ a_0 \oplus b_0 \oplus c_{in}] \end{aligned}$$

（数３）

$$\begin{aligned} g_i &= [c_{out}^{(i)}, S_i] \\ &= [a_ib_i \vee a_ic_{out}^{(i-1)} \vee b_ic_{out}^{(i-1)},\ a_i \oplus b_i \oplus c_{out}^{(i-1)}] \\ & (i = 1, 2, 3, 4, 5, 6, 7) \end{aligned}$$

すなわち、各分解関数は、２つの入力変数a_i, b_i及び桁上げを表す中間変数$c_{out}^{(i-1)}$の２を法とする和S_i及び桁上げの中間変数$c_{out}^{(i)}$を出力する論理関数となっている。

これを、第１図のような４段の論理関数メモリ４－０～４－３からなるプログラマブル論理デバイスで実現する。この場合には、第１２図に示したように、分解関数{g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7}を、４つの分解関数組{g_0, g_1}, {g_2, g_3}, {g_4, g_5}, {g_6, g_7}に分ける。そして、第１３図に示すように、各々の分解関数組を１つに合成し、４つの分解関数f_0, f_1, f_2, f_3で表す。ここで、各分解関数は、（数４）又は（数５）のような論理式で表される５入力３出力論理関数である。

（数４）

$$
\begin{aligned}
f_0 &= [c_{out}^{(0)}, S_1, S_0] \\
&= [a_1 b_1 \vee (a_1 \vee b_1)(a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), \\
&\quad a_1 \oplus b_1 \oplus (a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}),\ a_0 \oplus b_0 \oplus c_{in}]
\end{aligned}
$$

（数５）

$$
\begin{aligned}
f_i &= [c_{out}^{(i)}, S_{2i+1}, S_{2i}] \\
&= [a_{2i+1} b_{2i+1} \vee (a_{2i+1} \vee b_{2i+1})(a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), \\
&\quad a_{2i+1} \oplus b_{2i+1} \oplus (a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}),\ a_{2i} \oplus b_{2i} \oplus c_{out}^{(i-1)}] \\
&\quad (i = 1, 2, 3)
\end{aligned}
$$

各々の分解関数f_iの真理値表は、第１４図のようになる。そこで、第１５図の真理値表を、論理関数メモリ４－０の第０ページに記憶させる。第１６図の真理値表（ＬＵＴ）を論理関数メモリ４－１～４－３の第０ページに記憶させる。ここで、それぞれ、S_{2i}, S_{2i+1}, $c_{out}^{(i)}$ （$i \in \{1, 2, 3\}$）は、それぞれ接続回路５－（i+1）の入力線i_2, i_3, i_4に出力されるように論理関数メモリの出力ビットを割り当てる。

各入力選択メモリ３－０～３－３の第０ページには、それぞれ、０ビットシフト、５ビットシフト、９ビットシフト、１３ビットシフトの情報を記憶させる。すなわち、各入力選択メモリ３－０～３－３の入力選択情報（shf0, shf1, shf2, shf3）として、それぞれ、(0, 0, 0, 0)、(0, 1, 0, 1)、(1, 0, 0, 1)、(1, 1, 0, 1)を記憶させる。

各接続メモリ６－１～６－３の第０ページには、接続情報としてシフト情報（４ビット）を記憶させる。すなわち、接続情報として$(s_3, s_2, s_1, s_0)=(0,1,0,0)$を記憶させる。

以上のようにプログラムされた状態において、以下、プログラマブル論理デバイスの演算処理を説明する。

まず、入力変数レジスタ１には、変数A, B, c_{in}が入力される。入力変数レジスタ１はこれらの変数を記憶する。演算制御部１０は、入力選択メモリ３－０～３－３、論理関数メモリ４－０～４－３、及び接続メモリ６－１～６－３に対してページ選択番号p_rとして第０ページを設定する。

これにより、各入力選択メモリ３－０～３－３は、第０ページに書き込まれた入力選択情報（shf0, shf1, shf2, shf3）=(0,0,0,0)、(0,1,0,1)、(1,0,0,1)、(1,1,0,1)を、入力変数選択回路２－０～２－３の制御線shf0～shf3に出力する。入力変数選択回路２－０～２－３は、制御線shf0～shf3に入力された入力選択情報に従って、入力端子in(00)～in(16)の一部を出力端子out(00)～out(07)と電気的に接続する。

具体的には、入力選択メモリ３－０は、（shf0, shf1, shf2, shf3）=(0, 0, 0, 0)なので、in(00)～in(07)をout(00)～out(07)に接続する。入力選択メモリ３－１は、（shf0, shf1, shf2, shf3）=(0, 1, 0, 1)なので、５ビットシフトさせて、in(05)～in(12)をout(00)～out(07)に接続する。入力選択メモリ３－２は、（shf0, shf1, shf2, shf3）=(1, 0, 0, 1)なので、９ビットシフトさせて、in(09)～in(16)をout(00)～out(07)に接続する。入力選択メモリ３－３は、（shf0, shf1, shf2, shf3）=(1, 1, 0, 1)なので、１３ビットシフトさせて、in(13)～in(16)をout(00)～out(03)に接続する。このとき、out(04)～out(07)には入力変数は伝達されない。

また、接続メモリ６－１～６－３は、各々、第０ページに書き込まれた接続情報$(s_3^{(0)}, \cdots, s_0^{(0)}) = (0,1,0,0)$を接続回路５－１～５－３の各制御線に出力する。接続回

路５－１～５－３は、各制御線より入力される接続情報($s_3^{(k)}, \cdots, s_0^{(k)}$)に従って、入力線$i_8$～$i_{15}$と出力線$o_0$～$o_7$とを接続する。具体的には、各接続メモリ６－１～６－３は、４ビットシフトさせて、i_4～i_{11}をそれぞれo_0～o_7に接続し、i_{12}～i_{15}，i_0～i_3をそれぞれo_8～o_{15}に接続する。

次に、演算制御部１０は、入力変数レジスタ１により、入力変数Xの出力を行う。このとき、入力変数選択回路２－１～２－３の入力端子には、それぞれ、（表１）のように入力変数の値が入力される。

（表１）

i(00)	i(01)	i(02)	i(03)	i(04)	i(05)	i(06)	i(07)	i(08)
c_{in}	a_1	a_0	b_1	b_0	a_3	a_2	b_3	b_2
i(09)	i(10)	i(11)	i(12)	i(13)	i(14)	i(15)	i(16)	
a_5	a_4	b_5	b_4	a_7	a_6	b_7	b_6	

入力変数選択回路２－０は、(out(00)，out(01)，out(02)，out(03)，out(04))＝(c_{in}，a_1，a_0，b_1，b_0)を出力し、これを論理関数メモリ４－０に入力する。ここで、出力out(05)～out(07)は使用されないので省略した。

論理関数メモリ４－０は、第０ページに記憶されたＬＵＴ（LUT_0）のアドレス

($i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$)＝（$c_{in}, a_1, a_0, b_1, b_0$，－，－，－）に記憶された真理値

$f_0(c_{in}, a_1, a_0, b_1, b_0)$の値($o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7$)＝(－，－，$S_0$，$S_1$，$c_{out}^{(0)}$，－，－，－)を接続回路５－１に出力する。ここで、「－」はドントケア（０でも１でもよい。）を表す。このとき、接続回路５－１の各入力線（i_0，i_1，i_2，i_3，i_4，i_5，i_6，i_7）には(－，－，S_0，S_1，$c_{out}^{(0)}$，－，－，－)が入力される。

一方、入力変数選択回路２－１は、(out(00)，out(01)，out(02)，out(03))＝(a_3, a_2, b_3, b_2)を出力し、これを接続回路５－１の入力線（i_8, i_9, i_{10}, i_{11}）に入力する。

接続回路５－１からは、それぞれ、（表２）の値が出力される。

（表２）

o_0	o_1	o_2	o_3	o_4	o_5	o_6	o_7
$c_{out}^{(0)}$	-	-	-	a_3	a_2	b_3	b_2
o_8	o_9	o_{10}	o_{11}	o_{12}	o_{13}	o_{14}	o_{15}
-	-	-	-	-	-	S_0	S_1

接続回路５－１の出力線o_8～o_{15}は、外部出力線７－１として取り出される。従って、演算結果S_0, S_1が外部出力線７－１のうちの２本に出力される。

次に、論理関数メモリ４－１は、第０ページに記憶されたＬＵＴ（LUT_0）のアドレス（$i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$）＝（$c_{out}^{(0)}$, －, －, －, a_3, a_2, b_3, b_2）に記憶された真理値$f_1(c_{out}^{(0)}, a_3, a_2, b_3, b_2)$の値$(o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7)$＝(－, －, $S_2, S_3, c_{out}^{(1)}$, － , －, －)を接続回路５－２に出力する。このとき、接続回路５－２の各入力線（i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7）には(－, －, $S_2, S_3, c_{out}^{(1)}$, －, －, －)が入力される。

一方、入力変数選択回路２－２は、(out(00), out(01), out(02), out(03)) = (a_5, a_4, b_5, b_4)を出力し、これを接続回路５－２の入力線（i_8, i_9, i_{10}, i_{11}）に入力する。

接続回路５－２からは、それぞれ、（表３）の値が出力される。

（表３）

o_0	o_1	o_2	o_3	o_4	o_5	o_6	o_7
$c_{out}^{(1)}$	-	-	-	a_5	a_4	b_5	b_4
o_8	o_9	o_{10}	o_{11}	o_{12}	o_{13}	o_{14}	o_{15}
-	-	-	-	-	-	S_2	S_3

接続回路５－２の出力線o_8～o_{15}は、外部出力線７－２として取り出される。従って、演算結果S_2, S_3が外部出力線７－２のうちの２本に出力される。

次に、論理関数メモリ４－２は、第０ページに記憶されたＬＵＴ（LUT_0）のアドレス $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7) = (c_{out}^{(1)}, -, -, -, a_5, a_4, b_5, b_4)$ に記憶された真理値 $f_2(c_{out}^{(1)}, a_5, a_4, b_5, b_4)$ の値 $(o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7) = (-, -, S_4, S_5, c_{out}^{(2)}, -, -, -)$ を接続回路５－３に出力する。このとき、接続回路５－３の各入力線（i_0，i_1，i_2，i_3，i_4，i_5，i_6，i_7）には$(-, -, S_4, S_5, c_{out}^{(2)}, -, -, -)$が入力される。

一方、入力変数選択回路２－３は、(out(00)，out(01)，out(02)，out(03)) = (a_7，a_6，b_7，b_6)を出力し、これを接続回路５－３の入力線（i_8, i_9, i_{10}, i_{11}）に入力する。

接続回路５－３からは、それぞれ、（表４）の値が出力される。

（表４）

o_0	o_1	o_2	o_3	o_4	o_5	o_6	o_7
$c_{out}^{(2)}$	-	-	-	a_7	a_6	b_7	b_6
o_8	o_9	o_{10}	o_{11}	o_{12}	o_{13}	o_{14}	o_{15}
-	-	-	-	-	-	S_4	S_5

接続回路５－３の出力線o_8～o_{15}は、外部出力線７－３として取り出される。従って、演算結果S_4，S_5が外部出力線７－３のうちの２本に出力される。

最後に、論理関数メモリ４－３は、第０ページに記憶されたＬＵＴ（LUT_0）のアドレス $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7) = (c_{out}^{(2)}, -, -, -, a_7, a_6, b_7, b_6)$ に記憶された真理値 $f_3(c_{out}^{(2)}, a_7, a_6, b_7, b_6)$ の値 $(o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7) = (-, -, S_6, S_7, c_{out}^{(3)}, -, -, -)$ を出力線７－４に出力する。これにより、総ての演算結果が出力線７－１～７－４に出力されて演算が終了する。

〔例１終わり〕

（実施例２）

第１７図は本発明の実施例２に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、入力変数選択回路２－１～２－３の出力線の一部を、接続回路５－１～５－３を介すことなく、論理関数メモリ４－１～４－３の入力に直接接続したことを特徴とする。実用的な多くの論理関数においては、論理関数メモリ４－１～４－３の入力には、最低でも１つの入力変数が入力される場合が多い。すなわち、入力変数選択回路２－１～２－３の出力線の最低１本は、論理関数メモリ４－１～４－３の入力に接続される場合が多い。従って、汎用的な用途で使用するプログラマブル論理デバイスでは、最初から入力変数選択回路２－１～２－３の出力線の一部を、接続回路５－１～５－３を介さずに論理関数メモリ４－１～４－３の入力に直接接続しておけば、接続回路５－１～５－３の入力線の本数を減らすことができる。その結果、接続回路５－１～５－３を小型化することが可能となる。また、接続回路５－１～５－３を第４図のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化及び低消費電力化を図ることができる。

（実施例３）

第１８図は本発明の実施例３に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、接続回路５－１～５－３に入力する入力変数を選択する入力変数選択回路２－１～２－３に加えて、接続回路５－１～５－３を介さずに、論理関数メモリ４－１～４－３に直接入力する入力変数の選択を行う入力変数選択回路３０－１～３０－３及びその入力選択情報を記憶する入力選択メモリ３１－１～３１－３を設けたことを第１の特徴とする。また、前段の論理関数メモリ４－（ｉ－１）（ｉ∈｛１，２，３｝）から後段の論理関数メモリ４－ｉに入力される中間変数の一部を、接続回路５－ｉを介さずに直接入力するように構成したことを第２の特徴とする。

このように構成することで、接続回路５－１～５－３の入力線の本数を減らすことができる。その結果、接続回路５－１～５－３を小型化することが可能となる。また、接

続回路５－１～５－３を第４図のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化を図ることができる。

（実施例４）

第１９図は本発明の実施例４に係るプログラマブル論理デバイスの全体構成を表すブロック図、第２０図は第１９図の出力レジスタ及び出力デコーダの構成を表すブロック図、第２１図は第２０図の記憶素子の構成を表すブロック図である。本実施例においては、外部出力線７－１～７－４に出力された出力変数を一時的に記憶する出力変数レジスタ５１、及び出力変数レジスタ５１が出力変数を取り込むためのロード信号を出力する出力選択デコーダ５２を備えたことを特徴とする。他の構成については、第１図と同様であるため、説明は省略する。尚、本実施例においては各論理関数メモリ４－０～４－３は、内部にアドレス・ラッチ（図示せず）を備えており、クロックにより同期して動作するものとする。

出力変数レジスタ５１は、３２個の記憶素子Ｍ（ｉ，ｊ）（$i \in \{1,2,3,4\}$，$j \in \{0,1,2,3,4,5,6,7\}$）を備えている。各記憶素子Ｍ（ｉ，ｊ）は、データを保持するＤフリップ・フロップ（以下、「ＤＦＦ」という。）５３と２入力１出力のマルチプレクサ（以下、「ＭＵＸ」という。）５４とから構成されている。ＤＦＦ５３には、共通のクロック信号Clockが入力される。このクロック信号Clockが１のときに、ＤＦＦ５３はデータ入力Ｄに入力された値をラッチする。ＭＵＸ５４の出力はＤＦＦ５３のデータ入力Ｄに接続されている。また、ＤＦＦ５３の出力Ｑは、ＭＵＸ５４の０側の入力Ｄ０に接続されている。ＭＵＸ５４の１側の入力Ｄ１は、外部出力線７－ｉのｊ番目の線に接続されている。ＭＵＸ５４は、ロード信号Loadにより制御され、Load ＝ ０のときに０側の入力を選択し、Load ＝ １のときに１側の入力を選択する。

出力選択デコーダ５２には、演算制御部１０から、２ビットの出力選択信号ｔが入力される。そして、出力選択デコーダ５２は、４つの出力選択信号T_1～T_4を出力する

。t = (0, 0)のときは、(T1, T2, T3, T4) =(1, 0, 0, 0) が出力される。t = (0, 1) のときは、(T1, T2, T3, T4) =(0, 1, 0, 0) が出力される。t = (1, 0)のときは、(T1, T2, T3, T4) =(0, 0, 1, 0) が出力される。t = (1, 1)のときは、(T1, T2, T3, T4) =(0, 0, 0, 1)が出力される。各出力選択信号T_i（i ∈ {1, 2, 3, 4}）は、各記憶素子M（i, j）にロード信号Loadとして入力される。

演算制御部１０は、演算開始時には、t＝（００）として、論理関数メモリ４－０が演算結果を出力した後にt＝（０１）、論理関数メモリ４－１が演算結果を出力した後にt＝（１０）、論理関数メモリ４－２が演算結果を出力した後にt＝（１１）のように出力選択信号tを切り換えて、演算結果を出力変数レジスタ５１にラッチする。そして、演算が終了した時点で、出力変数レジスタ５１に記憶された出力変数を読み出すことで、演算結果を得ることができる。

（実施例５）

第２２図は本発明の実施例５に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、第１の出力選択回路２５、第２の出力選択回路２６、入力選択メモリ３－４、及び入力変数選択回路２－４を備えていることを特徴とする。尚、他の構成については、第１図と同様であるため、説明は省略する。

第２の出力選択回路２６は、入力変数選択回路２－４から入力される入力変数の値に従って、最終段の論理関数メモリ４－３から出力される出力変数のうちの全部又は一部を選択して出力する。尚、入力変数選択回路２－４は、第２図に示したものと同様のシフト回路で構成されている。但し、本実施例では、最終段の論理関数メモリ４－３の出力が８ビットであるため、入力変数選択回路２－４に使用されるシフト回路の出力は、３ビットとされる。一般には、入力変数選択回路２－４の出力は、最終段の論理関数メモリ４－３の出力の数Nに対して、

（数６）

$$\lceil \log_2 N \rceil$$

とされる。以下、入力変数選択回路２－４の出力out(00)～out(02)（第２図参照）を、それぞれ、F_0，F_1，F_2で表す。

第１の出力選択回路２５は、マルチプレクサにより構成されており、演算制御部１０から出力される出力選択信号ｔに従って、各論理関数メモリ４－０～４－２及び第２の出力選択回路２６から出力される出力変数のうちの何れかを選択して出力する。

ここで、出力選択信号ｔとは、第１の出力選択回路２５が選択する出力線束の指定番号（t∈{(00), (01), (10), (11)}）を表す２ビットの信号である。第１の出力選択回路２５は、ｔ＝(00)のときは接続回路５－１の出力線束（第４図のo_8～o_{15}）を選択し、ｔ＝(01)のときは接続回路５－２の出力線束を選択し、ｔ＝(10)のときは接続回路５－３の出力線束を選択し、ｔ＝(11)のときは第２の出力選択回路２６の出力線束を選択する。

第２３図は第２２図の第２の出力選択回路２６の構成を表すブロック図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

第２の出力選択回路２６は、第２３図に示したように、２入力１出力のマルチプレクサ（以下、「MUX」という。）３１～３７を多段にカスケード状に接続し、各段において出力を、MUX３８～MUX４４を通して取り出すことができるように構成されている。尚、第２３図では、説明の便宜上、出力線のビット数N_cが８ビットの例を示しているが、N_cは８ビットに限られるものではない。

第２３図において、第２の出力選択回路２６の入力線y_0～y_7は、論理関数メモリ４－３の出力に接続される。この入力線y_0～y_7より、出力変数Yの値が入力される。また、出力選択回路２６の出力線$f^{(0)}$～$f^{(7)}$からは、入力線y_0～y_7のうち選択された出力変数

Yの値($y_0, \cdots, y_{|Y|-1}$)が出力される。

入力線y_0, y_1、入力線y_2, y_3、入力線y_4, y_5、及び入力線y_6, y_7は、それぞれMUX３１、MUX３２、MUX３３、及びMUX３４の入力側に接続されている。MUX３１，３２、及びMUX３３，３４の出力は、それぞれMUX３５、及びMUX３６の入力側に接続されている。MUX３５，３６の出力は、MUX３７の入力に接続されている。

一方、（入力線y_0，MUX３１の出力）、（入力線y_2，MUX３２の出力）、（入力線y_4，MUX３３の出力）、及び（入力線y_6，MUX３４の出力）は、それぞれMUX３８、MUX３９、MUX４０、及びMUX４１の入力側に接続されている。また、（入力線y_1，MUX３５の出力）、（入力線y_3，MUX３７の出力）、及び（入力線y_5，MUX３６の出力）は、それぞれMUX４２、MUX４３、及びMUX４４の入力側に接続されている。

MUX３１～３４は、共通の入力変数F_0により切換制御がされる。すなわち、F_0が“０”のときは、MUX３１，３２，３３，３４は、それぞれ、入力線y_0, y_2, y_4, y_6を選択し、F_0が“１”のときは、MUX３１，３２，３３，３４は、それぞれ、入力線y_1, y_3, y_5, y_7を選択する。

MUX３５，３６は、共通の入力変数F_1により切換制御がされる。すなわち、F_1が“０”のときは、MUX３５，３６は、それぞれ、MUX３１，３３を選択し、F_1が“１”のときは、MUX３５，３６は、それぞれ、MUX３２，３４を選択する。

また、MUX３７は、入力変数F_2により切換制御がされる。すなわち、F_2が“０”のときは、MUX３７はMUX３５を選択し、F_2が“１”のときは、MUX３７はMUX３６を選択する。

MUX３１～３４によって、８本の入力線y_0～y_7のうちの４本が選択される。MUX３１～３６によって、８本の入力線y_0～y_7のうちの２本が選択される。MUX３１～３７によって、８本の入力線y_0～y_7のうちの１本が選択される。これは、目的論理関数$f(X)=(f_1(X), f_2(X), \cdots, f_m(X))$の出力変数の個数mが４個以下の場合、論理関数メモリ

４－３の出力する中間変数Yに対し、更に入力変数(F_0, F_1, F_2)による論理演算が可能であることを意味する。従って、論理関数メモリ４－０～４－３により行う演算における入力変数の数を減らすことができる。

すなわち、例えば、目的論理関数f($x_0, x_1, \cdots, x_{n-2}, x_{n-1}$)を、

(数７)

$$x_{n-1}f'(x_0, x_1, \cdots, x_{n-2}) \vee \overline{x}_{n-1}f'(x_0, x_1, \cdots, x_{n-2})$$

のようにシャノン展開して、関数f'($x_0, x_1, \cdots, x_{n-2}$)を４つの分解関数$f_0(X_0)$, $f_1(X_1, Y_1)$, $f_2(X_2, Y_2)$, $f_3(X_3, Y_3)$ (但し、$X_0 \cup X_1 \cup X_2 \cup X_3 = \{x_0, x_1, \cdots, x_{n-2}\}$) に関数分解して、最終段の分解関数$f_3(X_3, Y_3)$の出力数を４以下となるようにすれば、論理関数メモリ４－０～４－３に入力する入力変数の個数を１個減らすことができる。そして、入力変数x_{n-1}は、入力変数F_0として、第２の出力変数選択回路２６に入力するようにすればよい。

同様に、目的論理関数f($x_0, x_1, \cdots, x_{n-2}, x_{n-1}$)を、

(数８)

$$x_{n-1}x_{n-2}f''(x_0, x_1, \cdots, x_{n-3}) \vee x_{n-1}\overline{x}_{n-2}f''(x_0, x_1, \cdots, x_{n-3})$$
$$\vee \overline{x}_{n-1}x_{n-2}f''(x_0, x_1, \cdots, x_{n-3}) \vee \overline{x}_{n-1}\overline{x}_{n-2}f''(x_0, x_1, \cdots, x_{n-3})$$

のようにシャノン展開すれば、論理関数メモリ４－０～４－３に入力する入力変数の個数を２個減らすことができ、

(数９)

$$x_{n-1}x_{n-2}x_{n-3}f'''(x_0, x_1, \cdots, x_{n-4}) \vee x_{n-1}x_{n-2}\overline{x}_{n-3}f'''(x_0, x_1, \cdots, x_{n-4})$$
$$\vee x_{n-1}\overline{x}_{n-2}x_{n-3}f'''(x_0, x_1, \cdots, x_{n-4}) \vee x_{n-1}\overline{x}_{n-2}\overline{x}_{n-3}f'''(x_0, x_1, \cdots, x_{n-4})$$
$$\vee \overline{x}_{n-1}x_{n-2}x_{n-3}f'''(x_0, x_1, \cdots, x_{n-4}) \vee \overline{x}_{n-1}x_{n-2}\overline{x}_{n-3}f'''(x_0, x_1, \cdots, x_{n-4})$$
$$\vee \overline{x}_{n-1}\overline{x}_{n-2}x_{n-3}f'''(x_0, x_1, \cdots, x_{n-4}) \vee \overline{x}_{n-1}\overline{x}_{n-2}\overline{x}_{n-3}f'''(x_0, x_1, \cdots, x_{n-4})$$

のようにシャノン展開すれば、論理関数メモリ４－０～４－３に入力する入力変数の

個数を３個減らすことができる。

尚、MUX３１～MUX３７により選択された８ビット、４ビット、２ビット、又は１ビットの出力は、共通の８本の出力線によって第１の出力選択回路２５に出力する必要がある。そこで、第２３図では、MUX３８～４４によって、上記各本数の出力線が選択されたときの出力を、共通の出力線$f^{(0)}$～$f^{(7)}$を通して出力されるように構成されている。MUX３８～４４は、共通の選択切換変数Selectによって切換制御がされる。尚、選択切換変数Selectの設定値は、出力変数$f(X)=(f_1(X), f_2(X), \cdots, f_m(X))$の個数mによって、あらかじめ選択切換メモリ４５に記憶される。そして、演算制御部１０の制御によって、Selectに設定される。

MUX３８は、Selectが“０”のとき入力線y_0を選択し、Selectが“１”のときMUX３１の出力を選択し、選択した線の信号値を出力線$f^{(0)}$に出力する。MUX３９は、Selectが“０”のとき入力線y_2を選択し、Selectが“１”のときMUX３２の出力を選択し、選択した線の信号値を出力線$f^{(2)}$に出力する。MUX４０は、Selectが“０”のとき入力線y_4を選択し、Selectが“１”のときMUX３３の出力を選択し、選択した線の信号値を出力線$f^{(4)}$に出力する。MUX４１は、Selectが“０”のとき入力線y_6を選択し、Selectが“１”のときMUX３４の出力を選択し、選択した線の信号値を出力線$f^{(6)}$に出力する。

MUX４２は、Selectが“０”のとき入力線y_1を選択し、Selectが“１”のときMUX３５の出力を選択し、選択した線の信号値を出力線$f^{(1)}$に出力する。MUX４３は、Selectが“０”のとき入力線y_3を選択し、Selectが“１”のときMUX３７の出力を選択し、選択した線の信号値を出力線$f^{(3)}$に出力する。MUX４４は、Selectが“０”のとき入力線y_5を選択し、Selectが“１”のときMUX３６の出力を選択し、選択した線の信号値を出力線$f^{(5)}$に出力する。また、出力線$f^{(7)}$は、入力線y_7に直結されている。

目的論理関数fの演算結果f(X)のビット数（出力の本数）が８のときは、Selectを“０”とする。これにより、出力線$f^{(0)}$～$f^{(7)}$には、入力線y_0～y_7に入力される出力変数

Y=(y_0, …, y_7）の値が出力される。出力の本数が４のときは、Selectを“１”に設定し、入力変数F_0に“０”又は“１”を入力する。これにより、入力線y_0～y_7のうちF_0=0のときは（y_0，y_2，y_4，y_6）の４本が、F0=1のときは（y_1，y_3，y_5，y_7）の４本が、出力線$f^{(0)}$, $f^{(2)}$, $f^{(4)}$, $f^{(6)}$に出力される。出力の本数が２のときは、Selectを“１”とし、入力変数F_0、F_1に“０”又は“１”を設定する。これにより、(F_0，F_1)=(0，0)のときは、出力線$f^{(1)}$，$f^{(5)}$には入力線y_0～y_7のうち２本に入力される出力変数Y＝（y_0，y_4）の値が出力される。(F_0，F_1)=(0，0)，(0，1)，(1，1)のときは出力には(y_1，y_5)，(y_2，y_6)，(y_3，y_7)の値がそれぞれ出力される。出力の本数が１のときは、Selectを“１”とし、入力変数F_0，F_1，F_2に“０”又は“１”を設定する。これにより、出力線$f^{(3)}$には、入力線y_0～y_7のうち１本にF_0，F_1，F_2で指定した入力が出力される。

このように第２の出力選択回路２６を設けることにより、出力変数f(X)の個数が、最終段の論理関数メモリ４－３の全出力線の本数の１／２以下の場合には、第２の出力選択回路２６を用いて更に選択操作を行うことが可能である。これによって、論理関数メモリ４－０～４－３において演算を行う分解関数の入力変数の総数を実質的に１個以上増すことが可能となる。従って、プログラマブル論理デバイス全体で許容される入力変数の数を増やすことが可能となる。

尚、第２３図では、出力選択回路２６内のマルチプレクサは２入力１出力のものを使用しているが、一般に、ｗ入力１出力（ｗ≧２）のＭＵＸを使用することが可能である。

（実施例６）

第２４図は本発明の実施例６に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、第２の出力選択回路２６を、第１の出力選択回路２５の後段側に接続したことを特徴とする。尚、第１の出力選択回路２５及び第２の出力選択回

路２６の接続順序以外は、実施例５と同様に構成されているので、各部の説明については省略する。

接続回路５－１～５－３及び論理関数メモリ４－３から出力される変数組は、第１の選択回路２５において何れか一つが選択される。第１の選択回路２５において選択された変数組は、入力変数（F_1, F_2, F_3）及び選択切換変数Selectの値に基づいて、第２の選択回路２６によりその総て又は一部が選択され、出力変数として出力される。

ここで、任意のｎ変数論理関数$f(x_0, x_1, \cdots, x_{n-1})$は、以下の形で表現することが可能である。

（数１０）

$$f(x_0, x_1, \cdots, x_{n-3}, x_{n-2}, x_{n-1})$$
$$= x_{n-2}x_{n-1}f_0(x_0, x_1, \cdots, x_{n-3}) \vee x_{n-2}\overline{x}_{n-1}f_1(x_0, x_1, \cdots, x_{n-3})$$
$$\vee \overline{x}_{n-2}x_{n-1}f_2(x_0, x_1, \cdots, x_{n-3}) \vee \overline{x}_{n-2}\overline{x}_{n-1}f_3(x_0, x_1, \cdots, x_{n-3})$$

従って、上記部分関数f_0～f_3を論理関数メモリ４－０～４－３により実現し、各部分関数f_0～f_3とx_{n-2}, x_{n-1}等との積項を第２の出力選択回路２６により演算することで、ｎ変数の論理関数を実現することが可能となる。

（実施例７）

第２５図は本発明の実施例７に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本発明の実施例７に係るプログラマブル論理デバイスは、入力変数レジスタ１、入力変数選択回路２－０～２－３、入力選択メモリ３－０～３－３、論理関数メモリ４－０～４－３、接続回路５－０～５－３、接続メモリ６－０～６－３、領域指定メモリ８－０～８－３、及び演算制御部１０を備えている。尚、本実施例においては、各回路は共通のクロックによって同期して動作するものとする。

入力変数レジスタ１は、目的論理関数f(X)の演算に用いられるn個の入力変数X = (x_1, …, x_n)（但し、nは自然数）を記憶する。尚、入力変数Xは、外部入力線から入力変数レジスタ１に入力される。論理関数メモリ４－０～４－３は、目的論理関数fを関数分解して得られる分解関数{f_i；i∈{0, 1, 2, 3}}の真理値表を、ＬＵＴとして記憶する。尚、本実施例においては、一例として、論理関数メモリ４－i（i∈{０，１，２，３}）の段数を４段としているが、一般にはこの段数は任意に設定することができる。これらの論理関数メモリ４－０～４－３は、順序づけて配列され、それぞれ接続回路５－０～５－３を介してリング状に接続されている。

尚、本実施例における論理関数メモリ４－０～４－３は、第３図で説明したものと同様のものを使用する。その際、アドレス・デコーダ１６は、領域指定メモリ８－ｉから入力されるページ選択番号ｐ（p_r）に基づいて、メモリ領域FM_kのメモリ・アクセスを可能とする。

入力変数選択回路２－０～２－３は、入力変数レジスタ１から出力されるn個の入力変数X=(x_1, …, x_n)のうちから、それぞれ、各段の論理関数メモリ４－０～４－３に記憶された分解関数{f_i;i∈{0, 1, 2, 3}}の真理値表の入力変数X_i（i∈{0, 1, 2, 3}）を選択し、接続回路５－０～５－３へ出力する。入力選択メモリ３－０～３－３は、それぞれ、入力変数選択回路２－０～２－３の入力変数の選択に関する情報（以下、「入力選択情報」という。）が記憶されている。入力変数選択回路２－０～２－３は、各入力選択メモリ３－０～３－３から出力される入力選択信号に基づいて、入力変数の選択の切り換えを行う。

尚、本実施例においては、入力変数選択回路２－０～２－３には、第２図に示したようなシフタ回路を使用する。第２図の入力変数選択回路２－ｉ（i∈{0, 1, 2, 3}）において、入力端子in(00)～in(16)は、入力変数レジスタ１の出力端子に接続されている。従って、入力端子in(00)～in(16)からは入力変数X＝(x_1, …, x_n)が入力される。また

、入力変数選択回路２－ｉ（ｉ∈{0，1，2，3}）の出力端子out(00)～out(07)は、接続回路５－ｉの入力側端子の一部に接続されている。

各接続回路５－ｉ（ｉ∈{0，1，2，3}）は、論理関数メモリ４－（ｉ－1 mod 4）及び入力変数選択回路２－ｉより入力される、中間変数Y_i及び入力変数X_iを、順序を変えて、後段の論理関数メモリ４－ｉ及び外部出力線７－ｉへ接続する。各接続メモリ６－ｉ（ｉ∈{0，1，2，3}）は、各接続回路５－ｉの接続関係に関する情報（以下、「接続情報」という。）を記憶している。接続回路５－ｉは、接続メモリ６－ｉから出力される接続情報信号に基づき、接続関係の切り換えを行う。

尚、本実施例においては、説明の便宜上、接続回路５－０～５－３には、第４図に示したようなサイクリックなシフタ回路を使用することとする。第４図に示したような接続回路５－ｊ（ｊ∈{1，2，3}）において、入力線i_0～i_{15}のうち、i_0～i_7は前段の論理関数メモリ４－（ｊ－１）の出力側に接続されており、i_8～i_{15}は入力変数選択回路２－ｊの出力側に接続されている。また、接続回路５－０の入力線i_0～i_{15}のうち、i_0～i_7は前段の論理関数メモリ４－３の出力側に接続されており、i_8～i_{15}は入力変数選択回路２－０の出力側に接続されている。一方、接続回路５－ｊ（ｊ∈{0，1，2，3}）の出力線o_0～o_{15}のうち、o_0～o_7は、後段の論理関数メモリ４－ｊの入力側に接続されており、o_8～o_{15}は外部出力線７－ｊとなっている。

また、本実施例においては、各接続メモリ６－ｉ（ｉ∈{0，1，2，3}）は、第５図に示したものと同様のものが使用されている。

各領域指定メモリ８－ｉ（ｉ∈{0，1，2，3}）は、目的論理関数の演算段数に対応して、論理関数メモリ４－ｉに記憶されたＬＵＴのうちの使用するものが格納されているページ番号（以下、「領域指定変数」という。）が記憶されている。論理関数メモリ４－ｉは、領域指定メモリ８－ｉが出力する領域指定変数に従ってページを設定する。

演算制御部１０は、プログラマブル論理デバイス全体の演算処理の制御を行う。

第２６図は第２５図の演算制御部１０の構成を表すブロック図である。

演算制御部１０は、演算ステップ・レジスタ６１、ステップ・カウンタ６２、ページ・カウンタ６３、及び出力コントローラ６４を備えた構成からなる。

演算ステップ・レジスタ６１には、目的論理関数を関数分解した後の分解関数の数である演算ステップ数が格納される。ステップ・カウンタ６２は、現在演算が行われている分解関数の段数をカウントする。このステップ・カウンタ６２には、演算ステップ・レジスタ６１に格納された値が設定され、演算処理が１段進行するのに応じて、内部に格納された値を１ずつ減じていくダウン・カウンタによって構成されている。ステップ・カウンタ６２は、そのカウント値ｉが０となったときに、終了信号ENDを出力する。また、ステップ・カウンタ６２は、外部からリセット信号resetが入力されたとき、カウント値を演算ステップ・レジスタ６１に格納された値に設定する。

ページ・カウンタ６３は、入力選択メモリ３、接続メモリ６、及び領域指定メモリ８が出力するデータが格納されている各メモリのページ番号をカウントし、そのカウント値ｋをカウント信号pkとして出力する。ページ・カウンタ６３は、アップ・カウンタであり、出力コントローラ６４から出力されるチップ・イネーブル信号CE_3の立ち下がりを検出したときに、カウント値ｋを１だけ増加させる。尚、ページ・カウンタ６３は、リセット信号resetが入力されたとき、又はステップ・カウンタ６２から終了信号ENDが入力されたときに、カウント値ｋを０にリセットする。

出力コントローラ６４は、各々の論理関数メモリ４－０～４－３に対して、チップ・イネーブル信号CE_0～CE_3、及びアドレス・ストローブ信号$ADSP_0$～$ADSP_3$を出力することで、論理関数メモリ４－０～４－３の出力制御を行う。

尚、チップ・イネーブル信号CE_iは、ｉ段目の論理関数メモリ４－ｉを活性化する信号であり、チップ・イネーブル信号CE_iが“１”（真値）となった（アサートされた）ときに、論理関数メモリ４－ｉが外部からの入力信号の受け付け及びデータの出力を行う

ことが可能となる。アドレス・ストローブ信号$ADSP_i$は、論理関数メモリ４－ｉに対して、入力されたアドレスのラッチを制御するための信号である。論理関数メモリ４－ｉは、クロックエッジでアドレス・ストローブ信号$ADSP_i$が“１”（真値）となっている（アサートされている）ときに、入力線i(00)～i(07)に入力されているデータを、内部のアドレス・レジスタに設定する。また、アドレス・ストローブ信号$ADSP_i$が“１”から“０”（偽値）に遷移すると、内部のアドレス・レジスタに設定されているアドレス値をラッチする。尚、チップ・イネーブル信号CE_iが“０”（偽値）とされた状態では、$ADSP_i$が“１”となっても、入力線i(00)～i(07)に入力されているデータはラッチされない。

第２７図は第２６図の出力コントローラ６４の内部構成を表すブロック図である。

出力コントローラ６４は、４つのフリップ・フロップ（以下、「FF」という。）６５～６８からなるジョンソン・カウンタ６９、４つのAND回路７０～７３、及び１つのOR回路７４によって構成されている。

FF６５～６８のクロック端子Cには、共通のクロック信号clockが入力されている。また、FF６５，６６，６７，６８の出力Q_0，Q_1，Q_2，NOT(Q_3)は、それぞれ、FF６５，６６，６７，６８のデータ入力D_1，D_2，D_3，D_0に接続されている。また、OR回路７４は、リセット信号reset及び終了信号ENDの論理和のリセット信号reset’を出力する。FF６５～６８のリセット端子rstには、リセット信号reset’が入力されている。

このように構成することにより、ジョンソン・カウンタ６９の出力（Q_0，Q_1，Q_2，Q_3）の状態は、クロック信号clockが立ち上がるごとに、

（０，０，０，０）

→（１，０，０，０）

→（１，１，０，０）

→（１，１，１，０）

→（1，1，1，1）

→（0，1，1，1）

→（0，0，1，1）

→（0，0，0，1）

→（0，0，0，0）

のようにサイクリックに遷移する。すなわち、それぞれのクロックごとに、１ビット右にシフトして、最上位のビット（Q_3）を反転して、最下位のビット（Q_0）とするという動作を繰り返す。

FF６５，６７の出力Q_0，Q_2が、それぞれ、チップ・イネーブル信号CE_0，CE_1として出力される。また、FF６５，６７の出力NOT(Q_0)，NOT(Q_2)が、それぞれ、チップ・イネーブル信号CE_2，CE_3として出力される。ここで、NOT(Q_i)は、出力Q_iの反転出力である。

また、AND回路７０は、FF６５，６６の出力Q_0，NOT(Q_1)の論理積をアドレス・ストローブ信号$ADSP_0$として出力する。AND回路７１は、FF６７，６８の出力Q_2，NOT(Q_3)の論理積をアドレス・ストローブ信号$ADSP_1$として出力する。AND回路７２は、FF６５，６６の出力NOT(Q_0)，Q_1の論理積をアドレス・ストローブ信号$ADSP_2$として出力する。AND回路７３は、FF６７，６８の出力NOT(Q_2)，Q_3の論理積をアドレス・ストローブ信号$ADSP_3$として出力する。

尚、論理関数メモリ４－０～４－３は、チップ・イネーブル信号CE_0～CE_3が“０”とされた状態においては、低消費電力モードになるものとする。これにより、演算に使用されない論理関数メモリ４の消費電力が低減され、ＬＵＴカスケード回路全体を、より低消費電力化することができる。

以上のように構成された本実施例に係るプログラマブル論理デバイスにおいて、以下その動作について説明する。

第２８図は実施例７に係るプログラマブル論理デバイスの動作を表すフローチャー

ト、第２９図は実施例７に係るプログラマブル論理デバイスの動作時における各信号の変化を表すタイミング図である。

最初に、演算を行おうとする目的論理関数f(X)を関数分解した分解関数$\{f_0, \cdots, f_{s-1}\}$（1≦s）の真理値表を、ＬＵＴとして各論理関数メモリ４－０～４－３に書き込んでおく。ここで、f_i(i=0, 1, . , . , s-1)を表すＬＵＴは、４－(i mod 4)のメモリのint (i/4) ページに書き込んでおく。尚、同一の論理関数メモリ４において、異なるページに同一のＬＵＴが書き込まれることとなる場合には、複数ページに同一のＬＵＴを書き込むことはせず、１つのＬＵＴのみを書き込むようにする（以下、これを「書込縮約」という）。メモリ容量を節約するためである。また、入力選択メモリ３－０～３－３及び各接続メモリ６－１～６－３には、第０ページから順に上記各分解関数$\{f_0, \cdots, f_{s-1}\}$に対応した入力選択情報及び接続情報を書き込んでおく。また、領域指定メモリ８－ｉ（i∈{0, 1, 2, 3}）のｊページには、分解関数f_{i*4+j}（但し、0<=i*4+j<=s-1）に対応するＬＵＴが記憶されている論理関数メモリ４－ｉのページ数pを格納する。尚、各メモリへの書き込み機能については、第２５図では図示していないが、通常のメモリの書き込み方法により行われる。更に、演算制御部１０の演算ステップ・レジスタ６１には、分解関数の全段数sを書き込んでおく。尚、以下では、s>4であるものとして説明する。

以上の書き込みがされた状態において、まず、時間区間T_0において、演算制御部１０は、外部から入力されるリセット信号resetによって、ステップ・カウンタ６２、ページ・カウンタ６３のカウント値ｉ，pがそれぞれリセットされる（Ｓ１）。このとき、ステップ・カウンタ６２のカウント値ｉは、演算ステップ・レジスタ６１内に記憶された値sに初期化され、ページ・カウンタ６３のカウント値pは０に初期化される。また、当該リセット信号resetにより、演算制御部１０内の出力コントローラ６４のFF６５～６８がリセットされる（Ｓ２）。

このとき、FF６５～６８からなるジョンソン・カウンタ６９は、出力値として、（Q_0, Q_1, Q_2, Q_3）＝（0, 0, 0, 0），（$NOT(Q_0), NOT(Q_1), NOT(Q_2), NOT(Q_3)$）＝（1, 1, 1, 1）を出力する。従って、出力コントローラ６４は、各論理関数メモリ４－０，４－１，４－２，４－３に対して、チップ・イネーブル信号CE_0，CE_1，CE_2，CE_3として、それぞれ０，０，１，１を出力する。また、ページ・カウンタ６３は、カウント信号p_kとして０を、入力選択メモリ３－０～３－３、接続メモリ６－０～６－３、及び領域指定メモリ８－０～８－３に対して出力する（Ｓ３）。

これにより、入力選択メモリ３－ｉ（ｉ∈{0, 1, 2, 3}）は、第０ページに格納された入力選択情報を、入力変数選択回路２－ｉの制御線shf0～shf3に出力する。入力変数選択回路２－ｉは、制御線shf0～shf3に入力された入力選択情報に従って、入力端子in(00)～in(16)の一部を出力端子out(00)～out(07)と電気的に接続する。

また、接続メモリ６－ｉ（ｉ∈{0, 1, 2, 3}）は、第０ページに格納された接続情報（$s_{Nc-1}^{(0)}, \cdots, s_0^{(0)}$）を接続回路５－ｉに出力する。これにより、接続回路５－ｉは、各制御線より入力される接続情報（$s_3^{(0)}, \cdots, s_0^{(0)}$）に従って、入力線$i_8$～$i_{15}$と出力線$o_0$～$o_7$とを接続する。

更に、領域指定メモリ８－ｉ（ｉ∈{0, 1, 2, 3}）は、第０ページに格納された領域指定変数p(0)を論理関数メモリ４－ｉのアドレス・デコーダ１６に出力する。これにより、論理関数メモリ４－ｉは、第p(0)ページが選択され得る状態となる。

そして、入力変数レジスタ１は、各入力変数選択回路２－０～２－３に対して、入力変数の出力を開始する（Ｓ４）。これにより、接続回路５－０～５－３には、それぞれ、入力変数X_0，X_1，X_2，X_3が出力される。接続回路５－ｉ（ｉ∈{0, 1, 2, 3}）に入力された入力変数X_0，X_1，X_2，X_3は、設定された接続順序に従って、論理関数メモリ４－ｉの入力側に出力される。

続いて、時間区間T_1において、次のクロック信号clockの立ち上がりで、ジョンソン

・カウンタ６９の出力（Q_0, Q_1, Q_2, Q_3）は、（1, 0, 0, 0）に変化する。これにより、CE_0が“１”となり、論理関数メモリ４−０がアクセス可能な状態となるとともに、CE_2は“０”となる。従って、論理関数メモリ４−０から、データD_0が出力され始める（Ｓ５）。

また、これとほぼ同時に、AND回路７０の出力である$ADSP_0$が“１”となる。$ADSP_0$が“１”となると、論理関数メモリ４−０は、接続回路５−０から入力される入力変数X_0に基づいて、その内部のアドレス・レジスタが制定される（Ｓ６）。そして、論理関数メモリ４−０は、第０ページ内の制定されたアドレスに対応するＬＵＴの値（中間変数$Y_{4p+(s-i)}$）を、接続回路５−１に出力し始める。接続回路５−１に出力された中間変数$Y_{4p+(s-i)}$は、設定された接続関係に従って、論理関数メモリ４−１に入力される。また、場合によっては、中間変数$Y_{4p+(s-i)}$の一部は、外部出力線７−１へと伝達される。時間区間T_1の最後に$ADSP_0$が“１”から“0”に立ち下がり、論理関数メモリ４－０内部のアドレス・レジスタはラッチされる。

時間区間T_2において、$ADSP_0$が“１”となってから次のクロック信号clockが立ち上がったときに、ジョンソン・カウンタ６９の出力（Q_0，Q_1，Q_2，Q_3）は、（1, 1, 0, 0）に変化する。これにより、$ADSP_0$は“０”となる（Ｓ７）。これにより、論理関数メモリ４－０の出力データD_0の値が中間変数$Y_{4p+(s-i)}$に確定する。一方、このとき、Q_0=1なので、CE_0は“１”にラッチされている。

次いで、時間区間T_3において、次のクロック信号clockの立ち上がりで、ステップ・カウンタ６２は、カウント値ｉをデクリメントする（Ｓ８）。また、このとき、ジョンソン・カウンタ６９の値は(Q_0, Q_1, Q_2, Q_3)=(1, 1, 1, 0)となる。

ここで、ステップ・カウンタ６２のカウント値ｉが０でない場合において（Ｓ９）、出力Q_2の値が１に遷移すると、CE_1が“１”となるとともに、CE_3が“０”となる。それに伴って、論理関数メモリ４−１の出力線から出力データD_1が出力され始める（Ｓ１

０）。

また、これとほぼ同時に、AND回路７１の出力である$ADSP_1$が“１”となる（Ｓ１１）。これにより、論理関数メモリ４－１は、接続回路５－１を介して前段の論理関数メモリ４－０から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ４－１の出力データD_1が変化し、第pkページ内の制定されたアドレスに対応するＬＵＴの値（中間変数$Y_{4p+(s-i)}$）が、接続回路５－２に出力され始める。

そして、時間区間T_4において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ６９の値は$(Q_0, Q_1, Q_2, Q_3)=(1, 1, 1, 1)$となる。これにより、AND回路７１の出力である$ADSP_1$が“０”となり、論理関数メモリ４－１の出力データ$D_1$が、中間変数$Y_{4p+(s-i)}$に確定する（Ｓ１２）。接続回路５－２に出力された中間変数 $Y_{4p+(s-i)}$は、設定された接続関係に従って、論理関数メモリ４－２に入力される。また、場合によっては、中間変数$Y_{4p+(s-i)}$の一部は、外部出力線７－２へと伝達される。

次いで、時間区間T_5において、次のクロック信号clockの立ち上がりで、ステップ・カウンタ６２は、カウント値ｉをデクリメントする（Ｓ１３）。また、このとき、ジョンソン・カウンタ６９の値は$(Q_0, Q_1, Q_2, Q_3)=(0, 1, 1, 1)$となる。

ここで、ステップ・カウンタ６２のカウント値ｉが０でない場合（Ｓ１４）、出力Q_0の値が０に遷移すると、CE_2が“１”となるとともに、CE_0は“０”となる。それに伴って、論理関数メモリ４－０の出力が停止するとともに、論理関数メモリ４－２の出力線から出力データD_2が出力され始める（Ｓ１５）。

また、これとほぼ同時に、AND回路７２の出力である$ADSP_2$が“１”となる（Ｓ１６）。これにより、論理関数メモリ４－２は、接続回路５－２を介して前段の論理関数メモリ４－１から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ４－２の出力データD_2が変化し、第pk

ページ内の制定されたアドレスに対応するＬＵＴの値（中間変数$Y_{4p+(s-i)}$）が、接続回路５－３に出力され始める。

そして、時間区間T_6において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ６９の値は$(Q_0, Q_1, Q_2, Q_3)=(0, 0, 1, 1)$となる。これにより、AND回路７２の出力である$ADSP_2$が“０”となり、論理関数メモリ４－２の出力データ$D_2$が、中間変数$Y_{4p+(s-i)}$に確定する（Ｓ１７）。接続回路５－３に出力された中間変数$Y_{4p+(s-i)}$は、設定された接続関係に従って、論理関数メモリ４－３に入力される。また、場合によっては、中間変数$Y_{4p+(s-i)}$の一部は、外部出力線７－３へと伝達される。

次いで、時間区間T_7において、次のクロック信号clockの立ち上がりで、ステップ・カウンタ６２は、カウント値ｉをデクリメントする（Ｓ１８）。また、このとき、ジョンソン・カウンタ６９の値は$(Q_0, Q_1, Q_2, Q_3)=(0, 0, 0, 1)$となる。

ここで、ステップ・カウンタ６２のカウント値ｉが０でない場合において（Ｓ１９）、出力Q_2の値が０に遷移すると、CE_3が“１”となるとともに、CE_1は“０”となる。それに伴って、論理関数メモリ４－１の出力が停止するとともに、論理関数メモリ４－３の出力線から出力データD_3が出力され始める（Ｓ２０）。

また、これとほぼ同時に、AND回路７３の出力である$ADSP_3$が“１”となる（Ｓ２１）。これにより、論理関数メモリ４－３は、接続回路５－３を介して前段の論理関数メモリ４－２から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ４－３の出力データD_3が変化し、第pkページ内の制定されたアドレスに対応するＬＵＴの値（中間変数$Y_{4p+(s-i)}$）が、接続回路５－０に出力され始める。

そして、時間区間T_0において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ６９の値は$(Q_0, Q_1, Q_2, Q_3)=(0, 0, 0, 0)$となる。これにより、AND回路７３の出力である$ADSP_3$が“０”となり、論理関数メモリ４－３の出力データ$D_3$が、中間変数

$Y_{4p+(s-i)}$に確定する（Ｓ２２）。接続回路５－０に出力された中間変数$Y_{4p+(s-i)}$は、設定された接続関係に従って、論理関数メモリ４－０に入力される。また、場合によっては、中間変数$Y_{4p+(s-i)}$の一部は、外部出力線７－０へと伝達される。

このとき、ページ・カウンタ６３は、カウント値pをインクリメントする（Ｓ２３）。これにより、カウント信号p_pの値が１だけ増加し、入力選択メモリ３－０～３－３、接続メモリ６－０～６－３、及び領域指定メモリ８－０～８－３の選択されるページが切り替わる。これに伴って、入力変数選択回路２－０～２－３及び接続回路５－０～５－３の接続関係も切り替わる。尚、領域指定メモリ８－０～８－３の選択されるページが切り替わった場合であっても、各論理関数メモリは、チップ・イネーブル信号CE_i及びアドレス・ストローブ信号$ADSP_i$が“１”となってアドレスを受け付ける状態とならない限りにおいては、ページが切り替わることはない。

次いで、時間間隔T_1において、次のクロック信号clockの立ち上がりで、ステップ・カウンタ６２は、カウント値ｉをデクリメントする（Ｓ２４）。また、このとき、ジョンソン・カウンタ６９の出力Q_0の値が１となる。

ここで、ステップ・カウンタ６２のカウント値ｉが０でない場合には（Ｓ２５）、再びステップＳ５の動作に戻る。

上記ステップＳ９，Ｓ１４，Ｓ１９，Ｓ２５において、ステップ・カウンタ６２のカウント値ｉが０であった場合は、ステップ・カウンタ６２は終了信号ENDを出力する。これにより、演算制御部１０は演算動作を停止し、演算が終了する。

以上のように、演算処理は、前段から後段に向かって直列処理により行われる。そして、最後段の論理関数メモリ４－３における演算処理が終了した時点で、まだ総ての分解関数の演算が終了していない場合には、論理関数メモリ４－３から出力される中間変数を再び最前段の論理関数メモリにフィード・バックして、演算処理を繰り返す。そして、目的論理関数ｆの演算結果は、各外部出力線７－１～７－４より取り出される

。

次に、上記動作をより分かりやすく説明するため、具体的な例を用いてプログラマブル論理デバイスの具体的な動作を説明する。

〔例２〕加算回路

ここでは、簡単な例として、二つの２ｎビット（ｎ≧５）の二進数A=(a_{2n-1}, a_{2n-2}, …, a_1, a_0), B=(b_{2n-1}, b_{2n-2}, …, b_1, b_0)の加算を行う加算器を、上記プログラマブル論理デバイスによって実行する例を説明する。尚、下位からの桁上げ（キャリー）入力ビットとしてc_{in}も考慮して、以下のような演算を行う加算器を考える。

（数１１）

$$\begin{array}{cccccc} & a_{2n-1} & a_{2n-2} & \cdots & a_1 & a_0 \\ & b_{2n-1} & b_{2n-2} & \cdots & b_1 & b_0 \\ +) & & & & & c_{in} \\ \hline c_{out} & S_{2n-1} & S_{2n-2} & \cdots & S_1 & S_0 \end{array}$$

ここで、桁上げ入力ビットc_{in}とは、２ｎビット以上の大きな数の加算を行う場合に、加算器を直列につなげて使用する際に、下位の加算器からの桁上げを表すビットである。桁上げ出力ビットc_{out}も、同様に、上位の加算器に出力する桁上げを表すビットである。

尚、この例においては、入力変数選択回路２－０～２－３は、第２図と同様に、$\log_2(2n+1)$（$\log_2(2n+1)$が整数でないときは、$\log_2(2n+1)$を切り上げた数）段のシフタが直列接続された、(2n+1)入力８出力のシフタ回路で構成されているものとする。

このような二つの２ｎビットの二進数A,Bの加算を行う論理関数をf=f(X)（X=(A,B)）とすると、論理関数fは第３０図により表される。この関数は、第３１図のように、２ｎ個の分解関数{g_0, g_1, …, g_{2n-1}}に関数分解することが可能である。ここで、各分解関数は、（数１２）又は（数１３）のような論理式で表される。

(数１２)

$$g_0 = [c_{out}^{(0)}, S_0]$$
$$= [a_0b_0 \vee a_0c_{in} \vee b_0c_{in},\ a_0 \oplus b_0 \oplus c_{in}]$$

(数１３)

$$g_i = [c_{out}^{(i)}, S_i]$$
$$= [a_ib_i \vee a_ic_{out}^{(i-1)} \vee b_ic_{out}^{(i-1)},\ a_i \oplus b_i \oplus c_{out}^{(i-1)}]$$
$$(i = 1, 2, \cdots, 2n-1)$$

すなわち、各分解関数は、２つの入力変数a_i，b_i及び桁上げを表す中間変数$c_{out}^{(i-1)}$の２を法とする和S_i及び桁上げの中間変数$c_{out}^{(i)}$を出力する論理関数となっている。

これを、第２５図のような４段の論理関数メモリ４－０～４－３からなるプログラマブル論理デバイスで実現する場合には、第３２図に示したように、分解関数{g_0，g_1，…，g_{2n-1}}を、ｎ個の分解関数組{g_0，g_1}，{g_2，g_3}，…，{g_{2n-2}，g_{2n-1}}に分ける。そして、第３３図に示したように、各々の分解関数組を１つにまとめてｎ個の分解関数f_0，f_1，…，f_{n-1}で表す。ここで、各分解関数は、（数１４）又は（数１５）のような論理式で表される３変数入力３変数出力論理関数である。

(数１４)

$$f_0 = [c_{out}^{(0)}, S_1, S_0]$$
$$= [a_1b_1 \vee (a_1 \vee b_1)(a_0b_0 \vee a_0c_{in} \vee b_0c_{in}),$$
$$a_1 \oplus b_1 \oplus (a_0b_0 \vee a_0c_{in} \vee b_0c_{in}),\ a_0 \oplus b_0 \oplus c_{in}]$$

(数１５)

$$f_i = [c_{out}^{(i)}, S_{2i+1}, S_{2i}]$$
$$= [a_{2i+1}b_{2i+1} \vee (a_{2i+1} \vee b_{2i+1})(a_{2i}b_{2i} \vee a_{2i}c_{out}^{(i-1)} \vee b_{2i}c_{out}^{(i-1)}),$$
$$a_{2i+1} \oplus b_{2i+1} \oplus (a_{2i}b_{2i} \vee a_{2i}c_{out}^{(i-1)} \vee b_{2i}c_{out}^{(i-1)}),\ a_{2i} \oplus b_{2i} \oplus c_{out}^{(i-1)}]$$
$$(i = 1, 2, \cdots, n-1)$$

各々の分解関数f_iの真理値表は、第３４図のようになる。そこで、第３５図の真理値表を、論理関数メモリ４－０の第１ページに記憶させ、第３６図の真理値表（ＬＵＴ）を論理関数メモリ４－０～４－３の第０ページに記憶させる。ここで、それぞれ、S_{2i}，S_{2i+1}，$c_{out}^{(i)}$（$i \in \{1, 2, \cdots, n-1\}$）は、それぞれ接続回路５－（i+1）の入力線$i_2, i_3, i_4$に出力されるように論理関数メモリの出力ビットを割り当てる。

各入力選択メモリ３－０～３－３の第０ページには、それぞれ、０ビットシフト、５ビットシフト、９ビットシフト、１３ビットシフト、第pページ（$1 \leqq p \leqq [(n-1)/4]$）には、それぞれ、16p+1ビットシフト、16p+5ビットシフト、16p+9ビットシフト、16p+13ビットシフトの情報を記憶させておく。

接続メモリ６－０の第０ページには、接続情報として、８ビット分シフトさせる情報を記憶させる。接続メモリ６－１～６－３の第０ページには、接続情報として４ビット分シフトさせる情報を記憶させる。また、各接続メモリ６－０～６－３の第ｋページ（$1 \leqq p \leqq [(n-1)/4]$）には、接続情報として４ビット分シフトさせる情報を記憶させる。但し、接続メモリ６－α（$\alpha = n \bmod 4$）の第［n/4］ページの接続情報は、最終段の分解関数の出力変数（５ビット）を出力させるための接続情報なので、当該ページには５ビット分シフトさせる情報を記憶させる。尚、４ビット分シフトさせる場合には接続情報として$(s_3, s_2, s_1, s_0) = (0, 1, 0, 0)$を、５ビット分シフトさせる場合には$(s_3, s_2, s_1, s_0) = (0, 1, 0, 1)$を、８ビット分シフトさせる場合には接続情報として$(s_3, s_2, s_1, s_0) = (1, 0, 0, 0)$を接続メモリの各ページに記憶させる。例えば、ｎ＝１３の場合、接続メモリ６の内容は、以下の（表５）のようになる。

(表５)

Page	Memory for Interconnection 6-0	Memory for Interconnection 6-1	Memory for Interconnection 6-2	Memory for Interconnection 6-3
0	(1,0,0,0)	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)
1	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)
2	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)
3	(0,1,0,0)	(0,1,0,1)		

領域指定メモリ８－０の第０ページには、領域指定変数として１を記憶させておく。

また、領域指定メモリ８－１～８－３の第０ページには、領域指定変数として０を記憶させておく。更に、領域指定メモリ８－０～８－３の第ｋページ（１≦p≦［(n-1)/4］）には、領域指定変数として０を記憶させておく。例えば、ｎ＝１３の場合、接続メモリ６の内容は、以下の（表６）のようになる。

(表６)

Page	Memory for Interconnection 6-0	Memory for Interconnection 6-1	Memory for Interconnection 6-2	Memory for Interconnection 6-3
0	1	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0			

以上のようにプログラムされた状態において、以下、プログラマブル論理デバイスの演算処理を説明する。尚、以下の説明では、簡単のためｎ＝５として説明する。

まず、入力変数レジスタ１には、変数A，B，c_{in}が入力され記憶される。また、演算ステップ・レジスタ６１には、ｎ＝５が記憶される。そして、演算制御部１０は、まず、ステップ・カウンタ６２のカウント値ｉをｎ＝５にリセットし、ページ・カウンタ６３のカウント値ｋを０にリセットする。更に、ジョンソン・カウンタ６９も０にリセットする。これにより、演算制御部１０のページ・カウンタ６３は、入力選択メ

モリ３－０～３－３、接続メモリ６－０～６－３、及び領域指定メモリ８－０～８－３に対してページ選択番号p_rとして第０ページを設定する。

また、入力変数選択回路２－０～２－３は、各々の制御線に入力された入力選択情報に従って、入力端子in(00)～in(x-1)のうちの８つを出力端子out(00)～out(07)と電気的に接続する。

具体的には、入力選択メモリ３－０は、in(00)～in(07)をout(00)～out(07)に接続し、入力選択メモリ３－１は、５ビットシフトさせて、in(05)～in(12)をout(00)～out(07)に接続し、入力選択メモリ３－２は、９ビットシフトさせて、in(09)～in(16)をout(00)～out(07)に接続し、入力選択メモリ３－３は、１３ビットシフトさせて、in(13)～in(20)をout(00)～out(07)に接続する。

また、接続メモリ６－０～６－３は、各々、第０ページに書き込まれた接続情報$(s_3^{(0)}, \cdots, s_0^{(0)})$を接続回路５－０～５－３の各制御線に出力する。接続回路５－０～５－３は、各制御線より入力される接続情報$(s_3^{(0)}, \cdots, s_0^{(0)})$に従って、入力線$i_8$～$i_{15}$と出力線$o_0$～$o_7$とを接続する。具体的には、接続回路５－０は、８ビットシフトさせて、i_8～i_{15}をそれぞれo_0～o_7に接続し、i_0～i_7をそれぞれo_8～o_{15}に接続する。また、接続メモリ６－１～６－３は、４ビットシフトさせて、i_4～i_{11}をそれぞれo_0～o_7に接続し、i_{12}～i_{15}，i_0～i_3をそれぞれo_8～o_{15}に接続する。

また、領域指定メモリ８－０～８－３は、各々、第０ページに書き込まれた領域指定変数を、論理関数メモリ４－０～４－３のアドレス・デコーダ１６に対して出力する。これにより、論理関数メモリ４－０は、１ページがアクセス可能な状態となり、論理関数メモリ４－１～４－３は、１ページがアクセス可能な状態となる。

次に、演算制御部１０は、入力変数レジスタ１により、入力変数Xの出力を行う。このとき、入力変数選択回路２－１～２－３の入力端子には、それぞれ、（表７）のように入力変数の値が入力される。

(表７)

i(00)	i(01)	i(02)	i(03)	i(04)	i(05)	i(06)	i(07)	i(08)
c_{in}	a_1	a_0	b_1	b_0	a_3	a_2	b_3	b_2
i(09)	i(10)	i(11)	i(12)	i(13)	i(14)	i(15)	i(16)	i(17)
a_5	a_4	b_5	b_4	a_7	a_6	b_7	b_6	a_9
i(18)	i(19)	i(20)						
a_{10}	b_9	b_{10}						

入力変数選択回路２－０は、(out(00), out(01), out(02), out(03), out(04)) = (c_{in}, a_1, a_0, b_1, b_0)を出力し、これを接続回路５－０の入力線（i_8, i_9, i_{10}, i_{11}, i_{12}）に入力する。尚、出力out(05)～out(07)は使用されないので省略した。接続回路５－０は、８ビットシフトさせて、これらを出力線（o_0, o_1, o_2, o_3, o_4）に出力する。このようにして、入力変数(c_{in}, a_1, a_0, b_1, b_0)が論理関数メモリ４－０に入力される。

次に、演算制御部１０の出力コントローラ６４は、CE_0を“１”に設定して論理関数メモリ４－０のデータ出力を行わせるとともに、$ADSP_0$を“１”に設定して、論理関数メモリ４－０内部のアドレス・レジスタに、アドレス(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7)= (c_{in}, a_1, a_0, b_1, b_0, −, −, −) を制定する。ここで、「−」はドントケア（０でも１でもよい。）を表す。アドレス制定後に、出力コントローラ６４は、$ADSP_0$を“０”とする。

アドレス制定後、論理関数メモリ４－０の出力端子には、第１ページに格納されたＬＵＴ（LUT_1）のアドレス（c_{in}, a_1, a_0, b_1, b_0, −, −, −）に記憶された真理値 f_0(c_{in}, a_1, a_0, b_1, b_0)の値（−, −, S_0, S_1, $c_{out}^{(0)}$, −, −, −)が現れ、これが接続回路５－１に出力される。すなわち、接続回路５－１の各入力線（i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7）には(−, −, S_0, S_1, $c_{out}^{(0)}$, −, −, −)が入力される。

一方、入力変数選択回路２－１は、(out(00), out(01), out(02), out(03)) = (a_3, a_2, b_3, b_2)を出力し、これを接続回路５－１の入力線（i_8, i_9, i_{10}, i_{11}）に入力する

。

接続回路５－１からは、それぞれ、（表８）の値が出力される。

（表８）

o_0	o_1	o_2	o_3	o_4	o_5	o_6	o_7
$c_{out}^{(0)}$	-	-	-	a_3	a_2	b_3	b_2
o_8	o_9	o_{10}	o_{11}	o_{12}	o_{13}	o_{14}	o_{15}
-	-	-	-	-	-	S_0	S_1

接続回路５－１の出力線o_8～o_{15}は、外部出力線７－１として取り出される。従って、演算結果S_0, S_1が外部出力線７－１のうちの２本に出力される。

次に、演算制御部１０のステップ・カウンタ６２は、そのカウント値ｉをｉ－１＝４に更新する。そして、演算制御部１０の出力コントローラ６４は、CE_1を“１”に設定して論理関数メモリ４－１のデータ出力を行わせるとともに、$ADSP_1$を“１”に設定して、論理関数メモリ４－１内部のアドレス・レジスタに、アドレス(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7)= ($c_{out}^{(0)}$, -, -, -, a_3, a_2, b_3, b_2）を制定する。アドレス制定後に、出力コントローラ６４は、$ADSP_1$を“０”とする。更に、０段目の論理関数メモリ４－０の出力値は既にアドレス・レジスタに取り込んだので、出力コントローラ６４は、CE_0を“０”として、論理関数メモリ４－０の出力を停止させる。

アドレス制定後、論理関数メモリ４－１の出力端子には、第０ページに格納されたＬＵＴ（LUT_0）のアドレス（$c_{out}^{(0)}$, -, -, -, a_3, a_2, b_3, b_2）に記憶された真理値 f_1 ($c_{out}^{(0)}$, a_3, a_2, b_3, b_2)の値(-, -, S_2, S_3, $c_{out}^{(1)}$, - , -, -)が現れ、これが接続回路５－２に出力される。すなわち、接続回路５－２の各入力線（i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7）には(-, -, S_2, S_3, $c_{out}^{(1)}$, -, -, -)が入力される。

一方、入力変数選択回路２－２は、(out(00), out(01), out(02), out(03)) = (a_5, a_4, b_5, b_4)を出力し、これを接続回路５－２の入力線（i_8, i_9, i_{10}, i_{11}）に入力する

。

接続回路５－２からは、それぞれ、（表９）の値が出力される。

（表９）

o_0	o_1	o_2	o_3	o_4	o_5	o_6	o_7
$c_{out}^{(1)}$	-	-	-	a_5	a_4	b_5	b_4
o_8	o_9	o_{10}	o_{11}	o_{12}	o_{13}	o_{14}	o_{15}
-	-	-	-	-	-	S_2	S_3

接続回路５－２の出力線o_8～o_{15}は、外部出力線７－２として取り出される。従って、演算結果S_2，S_3が外部出力線７－２のうちの２本に出力される。

以下同様にして、論理関数メモリ４－２、論理関数メモリ４－３においても演算が行われる。そして、論理関数メモリ４－２の演算結果S_4，S_5は外部出力線７－３のうちの２本に出力される。また、論理関数メモリ４－３の出力値(－，－，S_6，S_7，$c_{out}^{(3)}$，－，－，－)は、接続回路５－０の入力線i_0～i_7にフィード・バックされる。

次に、演算制御部１０のステップ・カウンタ６２は、そのカウント値ｉをｉ－１＝１に更新する。ここで、論理関数メモリ４－３における演算処理が終了したので、ページ・カウンタ６３は、そのカウント値ｋを１だけインクリメントし、そのカウント値ｋをカウント信号pkに出力する。

これに伴い、入力選択メモリ３、接続メモリ６、及び領域指定メモリ８のアクセスページが変更されるため、これらのメモリからの出力も変更される。具体的には、入力選択メモリ３－０から出力されるシフト量の情報が１７ビットシフトに変更される。また、接続メモリ５－０から出力される接続情報が(s_3，s_2，s_1，s_0)＝(0,1,0,0)に変更され、接続メモリ５－１から出力される接続情報が(s_3，s_2，s_1，s_0)＝(0,1,0,1)に変更される。更に、領域指定メモリ８－０が出力する領域指定変数p(p_p=1)の値が０に変更され、論理関数メモリ４－０は０ページがアクセス可能な状態となる。

また、これにより、論理関数メモリ４－３の演算結果S_6, S_7は外部出力線７－０のうちの２本に出力される。

そして、上述の場合と同様に、論理関数メモリ４－０において演算処理が行われ、論理関数メモリ４－０の出力値(－, －, S_8, S_9, $c_{out}^{(4)}$, －, －, －)は接続回路５－１の入力線i_0～i_7に入力される。接続回路５－１は、これを５ビットシフトして、(S_8, S_9, $c_{out}^{(4)}$)を出力線（o_{13}, o_{14}, o_{15}）に出力する。これにより、論理関数メモリ４－０の演算結果(S_8, S_9, $c_{out}^{(4)}$)は外部出力線７－１のうちの３本に出力される。以上で、総ての演算結果が出力線７－０～７－３に出力されて演算が終了する。

〔例２終わり〕

尚、上記〔例２〕においては、各段のＬＵＴに入力する入力変数X_0, X_1, …, X_{s-1}は互いに共通の要素を持たない（すなわち、$X_i \cup X_j = \phi$ （$i \neq j$））と仮定したが、本発明におけるプログラマブル論理デバイスは、入力変数X_0, X_1, …, X_{s-1}の何れか２つが共通の要素を有するような場合にも使用することが可能である。

（実施例８）

第３７図は本発明の実施例８に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、接続回路５－０～５－３に入力する入力変数を選択する入力変数選択回路２－０～２－３に加えて、接続回路５－０～５－３を介さずに、論理関数メモリ４－０～４－３に直接入力する入力変数の選択を行う入力変数選択回路８０－０～８０－３及びその入力選択情報を記憶する入力選択メモリ８１－１～８１－３を設けたことを第１の特徴とする。また、前段の論理関数メモリ４－（ｉ－１）（$i \in \{1, 2, 3\}$）から後段の論理関数メモリ４－ｉに入力される中間変数の一部を、接続回路５－ｉを介さずに直接入力するように構成したことを第２の特徴とする。

このように構成することで、接続回路５－０～５－３の入力線の本数を減らすことが

できる。その結果、接続回路５－０～５－３を小型化することが可能となる。また、接続回路５－０～５－３を第４図のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化を図ることができる。

（実施例９）

第３８図は本発明の実施例９に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、フィード・バック用の接続回路の入力側に、最後段の論理関数メモリ４－３の複数個の出力を一時的に記憶し、フィード・バック用の接続回路５－０に出力する中間変数レジスタ８２を備えていることを特徴とする。

実施例７においては、論理関数メモリ４－０～４－３として、クロックに同期して動作するシンクロナスなメモリを使用した。しかし、論理関数メモリ４－０～４－３は、シンクロナスなメモリには限られず、非同期に動作するメモリを使用することも可能である。

しかしながら、最終段の論理関数メモリ４－３の出力は、最前段の論理関数メモリ４－０の入力にフィード・バックされることから、論理関数メモリ４－０～４－３に非同期のメモリを使用した場合、場合によっては発振を起こすなど、予期しない動作が起こる可能性がある。

そこで、本実施例では、中間変数レジスタ８２を備えた構成とし、１演算ループの演算が終了するごとに、論理関数メモリ４－０～４－３の出力値を一旦確定させ、論理関数メモリ４－３が出力する中間変数の値を中間変数レジスタ８２に記憶させる。そして、次の演算ループに移り、中間変数レジスタ８２に記憶された中間変数を値の論理関数メモリ４－０に入力して演算処理を行うようにする。

これにより、論理関数メモリ４－０～４－３に非同期なメモリを使用した場合にも、発振すること等を防止できる。

（実施例１０）

第３９図は本発明の実施例１０に係るプログラマブル論理デバイスの全体構成を表す図である。本実施例のプログラマブル論理デバイスは、演算部８５とｓ個（ｓ≧２）の出力回路８６－１～８６－ｓにより構成されている。本実施例のプログラマブル論理デバイスでは、最大ｓ個の演算をパイプライン処理により行うことが可能である。従って、第３９図では出力回路８６をｓ個設けた構成としている。しかし、出力回路８６の数は、必要に応じて減らすこともできる。

第４０図は本発明の実施例１０に係るプログラマブル論理デバイスの演算部８５の構成を表すブロック図である。第４０図において、入力変数レジスタ１、入力変数選択回路２－０～２－（ｓ－１）（ｓ≧３）、入力選択メモリ３-０～３-（ｓ－１）、論理関数メモリ４-０～４-（ｓ－１）、接続回路５-０～５-（ｓ－１）、接続メモリ６-０～６-（ｓ－１）、外部出力線７－１～７－ｓ、及び演算制御部１０は第１図と同様であり、同記号を付している。

本実施形態に係るプログラマブル論理デバイスは、ｓ個（ｓ≧２）の論理関数メモリ４－ｉ（ｉ＝０，…，ｓ－１）を直列に接続した構成である。ｉ－１段目（ｉ∈｛１，…，ｓ－１｝）の論理関数メモリ４－（ｉ－１）と、ｉ段目の論理関数メモリ４－ｉとの間には、接続回路５－ｉ，メモリ・アドレス・レジスタ（memory address register：以下、「ＭＡＲ」という。）９０－ｉ，及びダイレクト・アクセス・セレクタ（direct access selector：以下、「ＤＡＳ」という。）９１－ｉがこの順に接続されている。また、最前段の論理関数メモリ４－０の入力側には、ＤＡＳ９１－０が接続されている。

入力選択回路２－０の出力線は、すべて最前段のＤＡＳ９１－０の入力線に接続されている。入力選択回路２－ｉ（ｉ∈｛１，…，ｓ－１｝）の出力線は、その一部が論理関数メモリ４－ｉの入力線の一部と接続され、その他はＭＡＲ９０－ｉの入力線

の一部に接続されている。

最前段の論理関数メモリ４－０の入力線は、その全部がＤＡＳ９１－０の出力線と接続されている。また、それ以外の論理関数メモリ４－ｉ（ｉ∈｛１，…，ｓ－１｝）の入力線は、全部がその前段に配置されたＤＡＳ９１－ｉの各出力線に接続されている。

最後段の論理関数メモリ４－（ｓ－１）の出力線は、その全部が外部出力線７－ｓに接続されている。また、それ以外の論理関数メモリ４－ｉ（ｉ∈｛０，…，ｓ－２｝）の出力線は、その全部が接続回路５－（ｉ＋１）の入力線の一部に接続されている。

各論理関数メモリ４－０～４－ｓは、電源制御端子ＰＷを備えている。電源制御端子ＰＷに１が入力されたときには、各論理関数メモリ４－ｉ（ｉ∈｛０，…，ｓ－１｝）はウェイク・アップ状態（wake-up mode）となる。ＰＷに０が入力されたときには、各論理関数メモリ４－ｉは、低消費電力状態（low-power mode）となる。各論理関数メモリ４－０～４－ｓの電源制御端子ＰＷには、それぞれ、外部から電源制御信号ϕ_0～ϕ_{s-1}が入力されている。この電源制御信号ϕ_0～ϕ_{s-1}によって、各論理関数メモリ４－０～４－ｓの電源制御が行われる。すべての論理関数メモリを使用してパイプライン処理により複数のジョブを実行するときは、この電源制御信号ϕ_0～ϕ_{s-1}には常時１が入力される。一方、それ以外の場合には、ジョブの流れに合わせて、電源制御信号ϕ_0～ϕ_{s-1}に多相クロックを入力する。これにより、演算処理を実行する論理関数メモリのみをウェイク・アップ状態とするとともに、それ以外のものは低消費電力状態とし、消費電力の節減が図られる。

接続回路５－ｉ（ｉ∈｛１，…，ｓ－１｝）の入力線のうち、入力選択回路２－０の出力線に接続されたもの以外のものは、論理関数メモリ４－（ｉ－１）の出力線に接続されている。また、接続回路５－ｉの出力線の一部は、ＭＡＲ９０－ｉに接続さ

れており、その他のものは、外部出力線７－ｉに接続されている。尚、接続回路５－ｉの詳細については後述する。

ＭＡＲ９０－ｉ（ｉ∈｛１，…，ｓ－１｝）は、複数の入力線（データ入力線）から入力される変数値を一時的に保持する。ＭＡＲ９０－ｉは、中間変数レジスタとしての機能も有する。ＭＡＲ９０－ｉの入力線は、一部が入力選択回路２－ｉの出力線の一部と接続され、その他のものは接続回路５－ｉの出力線の一部と接続されている。ＭＡＲ９０－ｉの出力線は、その全部が、ＤＡＳ９１－ｉの入力線の一部に接続されている。

また、ＭＡＲ９０－ｉは、外部クロック入力線、リセット入力線、及びバイパス制御入力線を有する。外部クロック入力線には、外部からクロック信号（clock）が入力される。また、リセット入力線及びバイパス制御入力線には、演算制御部１０から、それぞれリセット信号（reset signal），バイパス制御信号が入力される。尚、このＭＡＲ９０－ｉは、論理関数メモリへの入力を取り込んで一時的に保持する中間変数レジスタとして機能する。この場合、外部クロック入力線に入力されるクロック信号がデータ・ストローブ信号となる。ＭＡＲ９０－ｉの詳細については後述する。

ＤＡＳ９１－ｉ（ｉ∈｛０，…，ｓ－１｝）は、外部から論理関数メモリ４－ｉに対して、データを直接書き込んだり、論理関数メモリ４－ｉのテストを行ったりするために設けられたものである。ＤＡＳ９１－ｉは、論理関数メモリ４－ｉの入力線（データ入力線）と同数の２入力１出力のマルチプレクサ（以下、「ＭＵＸ」という。）を備えている。各ＭＵＸの出力端子は、論理関数メモリ４－ｉの各入力線に接続されている。各ＭＵＸの２つの入力端子のうちの一方（０側入力端子）は、外部アドレス入力線１０１－ｉに接続されている。また、もう一方（１側入力端子）は、ＭＡＲ９０－ｉの出力線（一部は、直接、入力選択回路２－ｉの出力線）に接続されている。各ＭＵＸの選択制御端子は、選択制御線１０２が接続されている。選択制御線１０

２には、演算制御部１０から、ＤＡＳ選択制御信号（DAS select）が入力される。各ＭＵＸは、ＤＡＳ選択制御信号が０のときには０側入力端子を選択し、ＤＡＳ選択制御信号が１のときには１側入力端子を選択する。

外部アドレス入力線１０１－ｉには、論理関数メモリ４－ｉを直接アクセスする場合に、アクセスしようとするアドレスが入力される。ＤＡＳ選択制御信号を０として、外部アドレス入力線１０１－ｉにアドレスを入力することにより、論理関数メモリ４－ｉの直接アクセスが可能となる。この機能を利用して、論理関数メモリ４－ｉにデータを書き込んだり、論理関数メモリ４－ｉのテストを行ったりすることができる。

なお、本実施例においては、接続メモリ６－ｉ（ｉ∈｛１，…，ｓ－１｝）は、接続情報を表す接続変数のほかに、領域指定変数を記憶している。接続情報とは、二つの論理関数メモリ４－（ｉ－１），４－ｉ間において、前段の論理関数メモリ４－（ｉ－１）の出力線又は外部入力線のうち後段の論理関数メモリ４－ｉの各入力線に接続されるものを選択するための情報である。また、領域指定変数とは、論理関数メモリ４－（ｉ－１），４－ｉのメモリ領域を指定するための変数である。従って、接続メモリ６－ｉは、領域指定記憶手段としての機能も兼ね備えている。尚、これらの詳細については後述する。

第４１図は第４０図の接続回路とメモリ・アドレス・レジスタ（ＭＡＲ）の構成を表す図である。接続回路５－ｉは、メモリ・パッキング・シフタ（memory packing shifter：以下、「ＭＰＳ」という。）９２、及びレイル・セレクタ（rail selector）９３を備えている。

ＭＰＳ９２はk個の入力線（データ入力線）、入力線と同数（k個）の出力線（データ出力線）、及びシフト制御線を備えている。ＭＰＳ９２の各入力線には、前段の論理関数メモリ４の各出力線が接続されている。ＭＰＳ９２のシフト制御線には、接続

メモリ６－ｉから出力される領域指定変数Ｒの一部R_1（以下、「列選択変数」という。）が入力される。列選択変数R_1は、前段の論理関数メモリ４－（ｉ－１）内のメモリ・セルから読み出したデータの一部を取得する場合において、取り出すデータが格納されているメモリ・セルの列方向の領域を特定する変数である。ＭＰＳ９２は、論理関数メモリ４－（ｉ－１）の各出力線から入力される変数$Y_i=\{y_i\}$の順序を、列選択変数R_1により指定される量だけシフトして、出力線に出力する。これにより、ＭＰＳ９２は、前段の論理関数メモリ４－（ｉ－１）の出力変数から、必要なものを選択することができる。

レイル・セレクタ９３は、複数の２入力１出力のマルチプレクサ（以下、「ＭＵＸ」という。）９３ａを備えている。各ＭＵＸ９３ａは、１個の出力端子、１個の入力端子（データ入力端子）、及び１個の選択制御入力端子を備えている。

各ＭＵＸ９３ａの選択制御入力端子には、接続メモリ６－ｉから出力される各接続変数が入力される。各ＭＵＸ９３ａは、選択制御入力端子に入力される接続変数の値によって、０側入力端子又は１側入力端子を選択する。

各ＭＵＸ９３ａの出力端子は、ＭＡＲ９０－ｉの入力線に接続されている。各ＭＵＸ９３ａの一方の入力端子（０側入力端子）には、入力選択回路２－ｉの各出力線が接続されている。尚、入力選択回路２－ｉの出力線の一部は、ＭＵＸ９３ａを介さず、直接ＭＡＲ９０－ｉの入力線に接続されている。

これらのＭＵＸ９３ａは、ページ／入力変数選択群９３ｂと前段出力変数／入力変数選択群９３ｃとの２つの集合に分けられる。

ページ／入力変数選択群９３ｂの各ＭＵＸ９３ａの１側入力端子には、接続メモリ６－ｉから出力される領域指定変数Ｒの残りの一部R_2（以下、「行指定変数」という。）が入力される。行指定変数R_2は、後段の論理関数メモリ４－ｉ内のメモリ・セルにアクセスする場合において、アクセスするメモリ・セルの行方向の領域を特定する

変数である。ページ／入力変数選択群９３ｂの各ＭＵＸ９３ａは、行指定変数又は入力変数の何れか一方を選択して、ＭＡＲ９０－ｉの入力線に出力する。このように、ページ／入力変数選択群９３ｂの各ＭＵＸ９３ａを設けることによって、行指定変数のビット数を可変とすることができる。

前段出力変数／入力変数選択群９３ｃの各ＭＵＸ９３ａの１側入力端子には、ＭＰＳ９２の各出力線が接続されている。前段出力変数／入力変数選択群９３ｃの各ＭＵＸ９３ａは、前段の論理関数メモリ４－（ｉ－１）が出力する出力変数のうちＭＰＳにより選択されたもの、又は入力変数の何れか一方を選択して、ＭＡＲ９０－ｉの入力線に出力する。

ＭＡＲ９０－ｉは、同期型Ｄフリップ・フロップ（以下、「ＤＦＦ」という。）９０ａとバイパス選択回路９０ｂとの組が複数個配列された構成を有する。各ＤＦＦ９０ａは、データ入力端子（D）、データ出力端子（Q）、クロック端子（LOAD）、及びリセット端子（RST）を有する。また、各バイパス選択回路９０ｂは、２入力１出力のマルチプレクサで構成されている。

ＤＦＦ９０ａのクロック端子（LOAD）には、外部からのクロック信号（clock）が入力される。尚、このクロック信号が、データ・ストローブ信号となる。ＤＦＦ９０ａのリセット端子（RST）には、演算制御部１０が出力するリセット信号（reset）が入力される。

ＤＦＦ９０ａのデータ入力端子（D）は、バイパス選択回路９０ｂの１側入力端子と、バイパス線９０ｃにより接続されている。また、ＤＦＦ９０ａのデータ出力端子（Q）は、バイパス選択回路９０ｂの０側入力端子と接続されている。各ＤＦＦ９０ａのデータ入力端子は、前段の接続回路５－ｉの各出力線（レイル・セレクタ９３の各出力線又は入力変数選択回路２－ｉ）に接続されている。

バイパス選択回路９０ｂの選択制御入力端子には、演算制御部１０が出力するバイ

パス制御信号が入力される。バイパス選択回路９０ｂは、このバイパス制御信号により、０側入力端子又は１側入力端子を選択する。

ＤＦＦ９０ａのうちの一部は、そのデータ出力端子が後段のＤＡＳ９１－ｉに接続されており、残りの一部は、そのデータ出力端子が外部出力線７－ｉに接続されている。この場合、ページ／入力変数選択群９３ｂの各ＭＵＸ９３ａと接続されたＤＦＦ９０ａ、又は、入力変数選択回路２－ｉの出力線と直接接続されたＤＦＦ９０ａは、後段のＤＡＳ９１－ｉに接続される。前段出力変数／入力変数選択群９３ｃの各ＭＵＸ９３ａと接続されたＤＦＦ９０ａは、一部が後段のＤＡＳ９１－ｉに接続され、残りの一部は外部出力線７－ｉに接続される。

第４２図は本発明の実施例１０に係るプログラマブル論理デバイスの出力回路８６－ｉ（ｉ∈｛１，…，ｓ｝）の構成を表す図である。出力回路８６－ｉは、出力選択回路９４、出力シフタ９５、出力パッキング・シフタ９６、出力選択メモリ９７、出力パッキング・レジスタ９８、出力レジスタ９９を備えている。

出力選択回路９４は、演算制御部１０から出力されるステップ変数（step）の値に基づいて、外部出力線７－１～７－ｓに出力される出力変数Y_1～Y_sのうち何れか一つを選択する。そして、選択した出力変数Y_iを出力シフタ９５に出力する。出力シフタ９５には、出力選択回路９４から出力変数Y_iが入力される。出力シフタ９５は、出力変数Y_iをシフトして、出力パッキング・シフタ９６に出力する。

出力パッキング・シフタ９６は、複数のデータ入力線と複数のデータ出力線を有する。このデータ入力線の一部は、出力シフタ９５の出力線に接続されており、残りは、後述の出力パッキング・レジスタ９８の出力線に接続されている。出力パッキング・シフタ９６の出力線は、出力パッキング・レジスタの入力線に接続されている。出力パッキング・シフタ９６の入力線の数は、出力シフタ９５の出力線の数と出力パッキング・レジスタ９８の出力線の数の和だけある。一方、出力パッキング・シフタ９

６の出力線の数は、出力パッキング・レジスタ９８の出力線の数と同数である。

出力パッキング・シフタ９６は、入力線を所定の量だけシフトして、その一部を出力線に接続する。シフト量が０のときには、出力パッキング・レジスタ９８の出力線に接続された入力線が、出力パッキング・シフタ９６の出力線に接続されるように構成されている。

出力選択メモリ９７は、出力シフタ９５及び出力パッキング・シフタ９６のシフト量に関する情報が記憶されている。出力選択メモリ９７は、演算制御部１０から入力されるページ変数に従って、これらのシフト量情報を出力シフタ９５及び出力パッキング・シフタ９６に出力する。出力シフタ９５及び出力パッキング・シフタ９６は、このシフト量情報に従ってシフトを行う。

出力パッキング・レジスタ９８は、外部から入力されるクロックに従って、出力パッキング・シフタ９６の出力をラッチする。出力パッキング・レジスタ９８の出力は、出力レジスタ９９に出力されるとともに、出力パッキング・シフタ９６の入力側にフィード・バックされる。この構成により、外部出力線７０－ｉに順次出力される出力変数を、出力パッキング・レジスタ９８に順番に隙間なく詰め込むことができる。

出力レジスタ９９は、すべての演算が終了した時点で演算制御部１０から出力ロード信号（0_load）が入力されると、出力パッキング・レジスタ９８の出力を取り込んで保持する。

以上のように構成された本実施例に係るプログラマブル論理デバイスについて、以下その動作を説明する。

第４３図～第４５図は実施例１０に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。ここでは、論理関数メモリ４の段数が５段（ｓ＝５）の例について説明する。尚、ここでϕ_0～ϕ_sは５相クロックである。また、ＤＡＳ選択制御信号は１に設定されているものとする。従って、ＤＡＳ９１－０～９１－４は、前

段からの入力値を選択し、後段の論理関数メモリ４に伝達する。

初期状態では、すべての論理関数メモリ４－０～４－４は低消費電力状態とされている。

まず、第４３（ａ）図に示すように、外部入力線から入力変数Xが入力変数レジスタ１に入力される。入力変数レジスタ１は、入力変数Xを取り込んで保持する。入力変数Xは、入力変数レジスタ１から各入力変数選択回路２－０～２－４に出力される。各入力選択回路２－ｉ（ｉ∈｛０，１，２，３，４｝）は、入力選択メモリ３－ｉの出力値に従って、入力変数Xの中からｉ段目の論理関数メモリ４－ｉに入力する入力変数X_iを選択する。

入力選択メモリ２－０は、入力変数X_0を論理関数メモリ４－０の入力線に出力する。入力選択メモリ２－ｉ（ｉ∈｛１，２，３，４｝）は、入力変数X_iを接続回路５－ｉの入力線に入力する。

次に、第４３（ｂ）図に示すように、論理関数メモリ４－０の電源制御端子ＰＷに入力される電源制御信号ϕ_0が１になる。これにより、論理関数メモリ４－０は、ウェイク・アップ状態となる。そして、論理関数メモリ４－０の出力線から、１段目のＬＵＴによる演算結果が出力される。尚、第４３図～４５において、網掛けされた論理関数メモリは、ウェイク・アップ状態にある論理関数メモリを表している。

接続回路５－１は、接続メモリ６－１の出力に従って、論理関数メモリ４－０の出力線及び入力変数選択回路２－１の出力線とＭＡＲ９０－１の入力線とを接続する。そして、１段目のＬＵＴの演算結果の一部は、出力変数Y_1として外部出力線７－１に出力される。また、その他の変数は、中間変数U_1としてＭＡＲ９０－１の入力線に出力される。また、入力変数X_1もＭＡＲ９０－１の入力線に出力される。

次に、クロックclkの立ち上りで、ＭＡＲ９０－１は、中間変数U_1と入力変数X_1を取り込んで保持する。そして、電源制御信号ϕ_0が０となり、論理関数メモリ４－０は再

び低消費電力状態となる。それとともに、今度は、第４４（ａ）図に示すように、電源制御信号ϕ_1が１となり、論理関数メモリ４－１がウェイク・アップ状態となる。そして、論理関数メモリ４－１の出力線から、２段目のＬＵＴの演算結果が出力される。接続回路５－２は、接続メモリ６－２の出力に従って、論理関数メモリ４－１の出力線及び入力変数選択回路２－２の出力線とＭＡＲ９０－２の入力線とを接続する。そして、２段目のＬＵＴの演算結果の一部は、出力変数Y_2として外部出力線７－２に出力される。また、その他の変数は、中間変数U_2としてＭＡＲ９０－２の入力線に出力される。また、入力変数X_2もＭＡＲ９０－２の入力線に出力される。

以下、同様にして、第４４（ｂ）図，第４５（ａ）図，第４５（ｂ）図の順に演算が進行し、すべての演算が終了する。

このように、演算に使用する論理関数メモリのみをウェイク・アップ状態として演算を実行させ、使用していない他の論理関数メモリは低消費電力状態にすることにより、回路全体の消費電力を抑えることができる。

尚、ここでは、１つのタスクのみを実行させる動作について説明したが、本実施例のプログラマブル論理デバイスは、複数のタスクをパイプライン処理により実行することも可能である。これにより、多数のタスクを効率よく演算処理することが可能となる。

また、すべての論理関数メモリ４－０～４－（ｓ－１）をウェイク・アップ状態とし、ＭＡＲ９０－ｉ（ｉ∈｛１，…，ｓ－１｝）に入力するバイパス制御信号を１としておくことで、クロックを使用せずに非同期モードで高速に演算処理を行うことも可能である。

次に、領域指定変数ＲとＭＰＳ９２を使用した論理関数メモリ４－ｉ（ｉ∈｛０，…，ｓ－１｝.）のメモリ・パッキング方法に関して説明する。

第４６図はメモリ・パッキングの概念を説明する図である。第４６（ａ）図，第４

６（ｂ）図は、論理関数メモリのメモリ・マップを表している。

実施例１で示したプログラマブル論理デバイスでは、各論理関数メモリについて、ページを切り替えることにより複数の論理関数のＬＵＴを記憶させておくことが可能である。これを概念的に表示すると、第４６（ａ）図のようになる。第４６図では、ページ指定ビットとして、論理関数メモリの行アドレスの上位２ビットを割り当てている。そうすると、論理関数メモリは４ページに区分される。そして、ページ毎にＬＵＴを格納すると、最低４個のＬＵＴを格納することができる。また、１個のＬＵＴが行内のすべての列アドレスを使用していないような場合には、１つのページ内に複数のＬＵＴを格納することも可能である。

例えば、第４６（ａ）図では、２ページ目に、論理関数f_2，f_6及びf_7の３つのＬＵＴを格納している。また、３ページ目にはf_3及びf_8の２つのＬＵＴを格納している。

しかしながら、この方法では、論理関数メモリの内部に未使用の領域が散在することとなり、メモリ使用効率が低い。そこで、第４６（ｂ）図のように、論理関数メモリ内にできるだけ隙間なくＬＵＴを詰め込むようにすれば、メモリ使用効率が向上する。従って、１つの論理関数メモリ内に格納することのできるＬＵＴの数も増やすことができる。このように、論理関数メモリ内にＬＵＴをできるだけ隙間なく格納することを「メモリ・パッキング」という。

上記のようなメモリ・パッキングを実現するためには、ページ指定ビットの数を可変にする必要がある。また、論理関数メモリ内の列アドレスを、任意の位置から読み出せるようにすることが必要である。

第４１図に示した本実施例に係る接続回路５－ｉ（ｉ∈｛１，…，ｓ－１｝）は、このようなメモリ・パッキングを行うことを可能とする。まず、ページ指定は、接続メモリ６－ｉが出力する行指定変数R_2により行う。また、ページ指定ビットとして何

ビット使用するかは、接続メモリ６－ｉが出力する接続変数のうち、ページ／入力変数選択群９３ｂのＭＵＸ９３ａに入力するものにより指定される。第４１図の例では、ページ指定ビットとして最大３ビットまで使用することが可能である（尚、これは３ビットに限らず、目的にあわせて何ビットで設計してもよい）。ページ指定ビットとして使用しない場合には、代わりにそのビットには入力変数を入力することができる。

次に、論理関数メモリ内の列アドレスの読み出し位置の変更は、ＭＰＳ９２により行う。例えば、前段の論理関数メモリ４－（ｉ－１）の指定された行において、列方向に８ビットずらして読み出したい場合には、ＭＰＳ９２により論理関数メモリ４－（ｉ－１）の出力を８ビットシフトさせる。これにより、論理関数メモリ内の列アドレスを、任意の位置から読み出せるようにすることができる。なお、列アドレスの読み出し位置の指定は、接続メモリ６－ｉが出力する列選択変数R_1により行われる。

以上のように、本実施例の接続回路５－ｉ（ｉ∈｛１，…，ｓ－１｝）を使用すると、メモリ・パッキングを実現することが可能となる。

最後に、第４２図に示した出力回路８６－ｉの動作について簡単に説明する。まず、演算制御部１０は、出力変数Y_iが出力される論理関数メモリ４－（ｉ－１）の段数（ｉ－１）をステップ変数（step）として出力回路８６－ｉに出力する。出力セレクタ９４は、このステップ変数（step）の値に基づき、出力変数Y_iを選択する。選択された出力変数Y_iは、出力シフタ９５の入力線に出力される。

一方、出力選択メモリ９７は、ステップ変数（step）の値に基づき、出力シフタ９５及び出力パッキング・シフタ９６のシフト量に関する情報を出力する。

通常は、必ずしも出力変数Y_iのすべてのビットが利用されているわけではなく、その一部のビットのみに有効な出力変数が出力される。そこで、出力シフタ９５は、出力選択メモリ９７から入力されるシフト量情報に従って、出力変数Y_iをシフトさせる。こ

れにより、出力変数Y_iの未使用ビットを除いて、変数を端のビットから詰め込むことができる。出力シフタ９５の出力は、出力パッキング・シフタ９６の入力線に出力される。

出力パッキング・シフタ９６は、更に、入力線を所定の量だけシフトして、その一部を出力線に接続する。例えば、出力シフタ９５から出力される出力変数Y_iの有効なビット数がｒビットであったとする。この場合、出力パッキング・シフタ９６は、入力線をｒビットだけシフトして出力線に接続する。これにより、出力シフタ９５から出力されるｒビットの出力変数は、ちょうど出力パッキング・レジスタ９８の入力線の下位ｒビットの範囲に詰め込まれる。出力パッキング・レジスタ９８は、クロックclkに同期して、出力パッキング・シフタ９６の出力値を保持する。

次に、ステップ変数（step）がｉとなり、出力セレクタ９４が出力変数Y_{i+1}を選択すると、同様にして、出力パッキング・レジスタ９８の入力線の下位ビットに、出力変数が出力される。このとき、先に出力パッキング・レジスタ９８に保持された変数値は、出力パッキング・シフタ９６の入力線にフィード・バックされている。従って、先に出力パッキング・レジスタ９８に保持された変数値も、新たな出力変数と同じビット数だけシフトして出力パッキング・レジスタ９８に入力される。そして、出力パッキング・レジスタ９８は、クロックclkに同期して、出力パッキング・シフタ９６の出力値を保持する。

このようにして、出力パッキング・シフタ９６に出力変数が順次隙間なく詰め込まれる。そして、演算が終了した時点で、演算制御部１０は出力ロード信号（0_load）を１とする。これにより、出力レジスタ９９は出力パッキング・シフタ９６の出力値を取り込んで記憶し、外部に出力する。

このようにして、本実施例の出力回路を使用すれば、各論理関数メモリから個々に出力される演算結果をパッキングして、まとめて出力させることが可能となる。

（実施例１１）

第４７図は本発明の実施例１１に係るプログラマブル論理デバイスの構成を表す図である。本実施例のプログラマブル論理デバイスは、基本的な構成は実施例１０のプログラマブル論理デバイスと同様であるが、本実施例ではｓ個（ｓ≧２）の論理関数メモリ４－０～４－（ｓ－１）がリング状に接続されている点において実施例１０とは異なる。すなわち、論理関数メモリ４－（ｓ－１）の後段に、接続回路５－ｓ、ＭＡＲ９０－ｓ、ＤＡＳ９１－ｓが設けられており、ＤＡＳ９１－ｓの出力線は、論理関数メモリ４－０の入力線に接続されている。

このように、論理関数メモリ４－０～４－（ｓ－１）をリング状に接続したことで、このプログラマブル論理デバイスで実行可能なＬＵＴカスケードの段数をｓ段よりも大きくすることが可能である。従って、ＬＵＴカスケードの設計自由度が大きくなる。

例えば、ｓ＝２の場合を考える。この２つの論理回路メモリ４－０，４－１を用いて４段のＬＵＴカスケードを実現する。第４８図に示したように、１段目のＬＵＴを論理回路メモリ４－０の０ページ（第４８（ａ）図）、２段目のＬＵＴを論理回路メモリ４－１の０ページ（第４８（ｂ）図）、３段目のＬＵＴを論理回路メモリ４－０の１ページ（第４８（ｃ）図）、４段目のＬＵＴを論理回路メモリ４－１の１ページ（第４８（ｄ）図）に格納すればよい。尚、ここではページを固定したが、論理回路メモリのメモリ使用効率を上げるためには、上述したメモリ・パッキングの手法を用いることもできる。

また、この論理回路メモリをリング状に接続したプログラマブル論理デバイスを用いて、以下の〔例３〕に示すように、２つ以上の組み合わせ論理回路の演算を１つのプログラマブル論理デバイスで同時に実行することが可能となる。

〔例３〕

（数１６）、（数１７）で表される２つの論理関数ｆ，ｇを考える。

（数１６）

$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$

（数１７）

$$g = ((x_4x_5 \vee x_6)x_1 \vee x_2)x_3$$

（数１６）の論理関数ｆは、第４９（ａ）図に示したような６段のＬＵＴカスケードにより表現することができる。（数１７）の論理関数ｇは、第４９（ｂ）図に示したような６段のＬＵＴカスケードにより表現することができる。

そこで、この２つのＬＵＴカスケードを、６個の論理回路メモリをリング状に接続したプログラマブル論理デバイスを用いて実現する。この場合、第５０図に示したように、論理関数ｆのＬＵＴカスケードの各段の入力変数と、論理関数ｇのＬＵＴカスケードの各段の入力変数が重複するように、互いにずらして配置する。

そして、この２つのＬＵＴカスケードを、第５１図に示したような１つのＬＵＴリングに合成する。これにより、２つの組み合わせ論理回路が１つのＬＵＴリングにより表現される。そして、このＬＵＴリングを各論理関数メモリに格納すれば、２つの組み合わせ論理回路ｆ、ｇの演算を１つのプログラマブル論理デバイスで同時に実行することが可能となる。

〔例３終わり〕

産業上の利用可能性

以上のように、本発明によれば、接続回路と接続メモリを用いて、それぞれの目的論理関数に応じて、前段の論理関数メモリの出力線及び入力変数の入力線と後段の論理関数メモリの入力線との接続関係を再構成することが可能な構成とした。これにより、使用可能な論理関数のレイル数及び入力変数の個数の組み合わせの自由度が増す

。そして、より多くの目的論理関数の論理回路を１つのＬＵＴカスケード論理回路により設計することが可能となる。また、レイル数及び入力変数の数の組み合わせを最適化できるため、論理関数メモリの入力線数を極力少なくすることができる。そのため、メモリの使用効率も向上する。その結果、ＬＳＩチップの利用効率が向上する。従って、回路の小型化・高集積化を図ることができる。

また、接続回路が、前段の論理関数メモリの出力を、論理関数の演算結果を出力する外部出力線に接続可能とした。これにより、ＬＵＴカスケードの途中の論理関数メモリの出力を出力変数として取り出し、必要なメモリ量を削減し、演算速度を高速化することが可能となる。

また、各論理関数メモリについて、複数の目的論理関数に対して、論理関数のＬＵＴの異なるメモリ領域を割り当て、領域指定記憶手段の出力に従ってアクセス可能なメモリ領域を選択的に切り換えることで、複数の目的論理関数の演算を実行させることが可能となる。

更に、各接続回路及び各論理関数メモリに対して入力する入力変数を、各入力変数選択回路により個別に選択することができる。従って、複数の論理関数メモリの演算を同時に行うことが可能であり、パイプライン処理が可能となる。

請　求　の　範　囲

１．以下の構成を含むプログラマブル論理デバイス（programmable logic device）：

論理関数のＬＵＴ（look up table）を記憶するための、直列に順序づけて配列された論理関数メモリ；

前記各論理関数メモリに対する入力変数が入力される複数の外部入力線；

二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ；

二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

２．以下の構成を含むプログラマブル論理デバイス：

論理関数のＬＵＴを記憶するための、リング状に配列された論理関数メモリ；

前記各論理関数メモリに対する入力変数が入力される複数の外部入力線；

二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ；

二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

３．論理関数の演算結果を外部回路に出力するための外部出力線を備え；

前記接続メモリは、二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線のうち前記外部出力線に接続されるものを選択するための接続情報をも

記憶するものであり；

前記接続回路は、二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線と前記外部出力線との接続をも行うものである；

ことを特徴とする請求の範囲第１項又は第２項記載のプログラマブル論理デバイス。

４．論理関数メモリのメモリ領域を指定する領域指定変数を記憶する領域指定記憶手段を備え；

前記接続回路は、二つの前記論理関数メモリ間において、領域指定記憶手段の出力に従って、

前段の前記論理関数メモリの出力及び外部入力線からの入力変数が、領域指定変数により特定される後段の前記論理関数メモリのメモリ領域に入力されるように、

又は、領域指定変数により特定される前段の前記論理関数メモリのメモリ領域からの出力及び外部入力線からの入力変数が、後段の前記論理関数メモリに入力されるように、

前段の前記論理関数メモリの出力線、外部入力線、及び領域指定記憶手段の出力線と、後段の前記論理関数メモリの入力線との接続を行うものである；

ことを特徴とする請求の範囲第１項乃至第３項の何れか一記載のプログラマブル論理デバイス。

５．前記論理関数メモリの入力側又は出力側に設けられ、外部から入力されるデータ・ストローブ信号に従って、前記論理関数メモリへの入力又は前記論理関数メモリの出力を取り込んで一時的に保持する中間変数レジスタ；

を備えていることを特徴とする請求の範囲第１項乃至第４項の何れか一記載のプログラマブル論理デバイス。

６．前記中間変数レジスタと並列に接続されたバイパス線；

及び、前記中間変数レジスタの出力側に設けられ、前記中間変数レジスタの出力線又は前記バイパス線の何れか一方を選択し、選択された線の信号を出力するバイパス選択回路；

を備えていることを特徴とする請求の範囲第１項乃至第５項の何れか一記載のプログラマブル論理デバイス。

７．前記データ・ストローブ信号を計数し、演算を実行する前記論理関数メモリの番号を特定する論理関数メモリ特定手段；

を備えていることを特徴とする請求の範囲第５項又は第６項記載のプログラマブル論理デバイス。

８．演算処理を実行させる前記論理関数メモリを通常動作状態とし、それ以外の前記論理関数メモリを低消費電力状態とする制御を行う電源制御手段；

を備えている請求の範囲第５項乃至第７項の何れか一記載のプログラマブル論理デバイス。

９．前記各論理関数メモリの一部の入力線は、前記接続回路を介すことなく前記外部入力線に直接接続されていることを特徴とする請求の範囲第１項乃至第８項の何れか一記載のプログラマブル論理デバイス。

１０．前記各論理関数メモリの一部の出力線は、前記接続回路を介すことなくその後段の論理関数メモリの一部の入力線に直接接続されていることを特徴とする請求の範囲第１項乃至第９項の何れか一に記載のプログラマブル論理デバイス。

１１．前記接続回路は、複数のセレクタ回路を含み；

前記各セレクタ回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線と前記外部入力線とのうち何れか一つ、又は、前段の前記論理関数メモリの出力線と前記外部入力線と領域指定記憶手段の出力線とのうち何れか一つを選択して後段の前記論理関数メモリの入力線に接続するものであること；

を特徴とする請求の範囲第１項乃至第１０項の何れか一記載のプログラマブル論理デバイス。

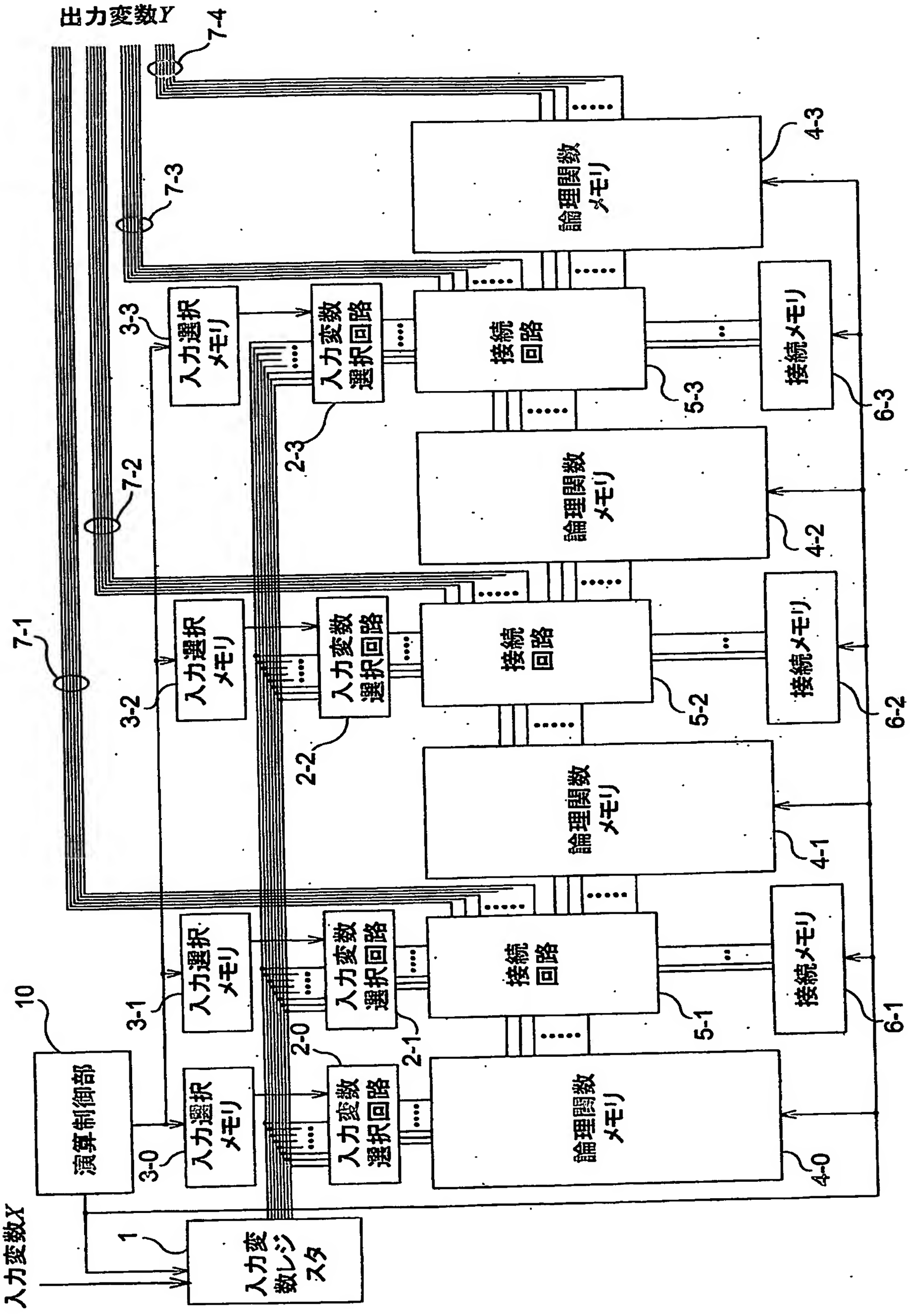
１２．前記接続回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線の接続順序をシフトして後段の前記論理関数メモリの入力線に接続するシフタ回路を含むことを特徴とする請求の範囲第１項乃至第１１項の何れか一記載のプログラマブル論理デバイス。

１３．前記接続回路は、複数のマルチプレクサを含み；

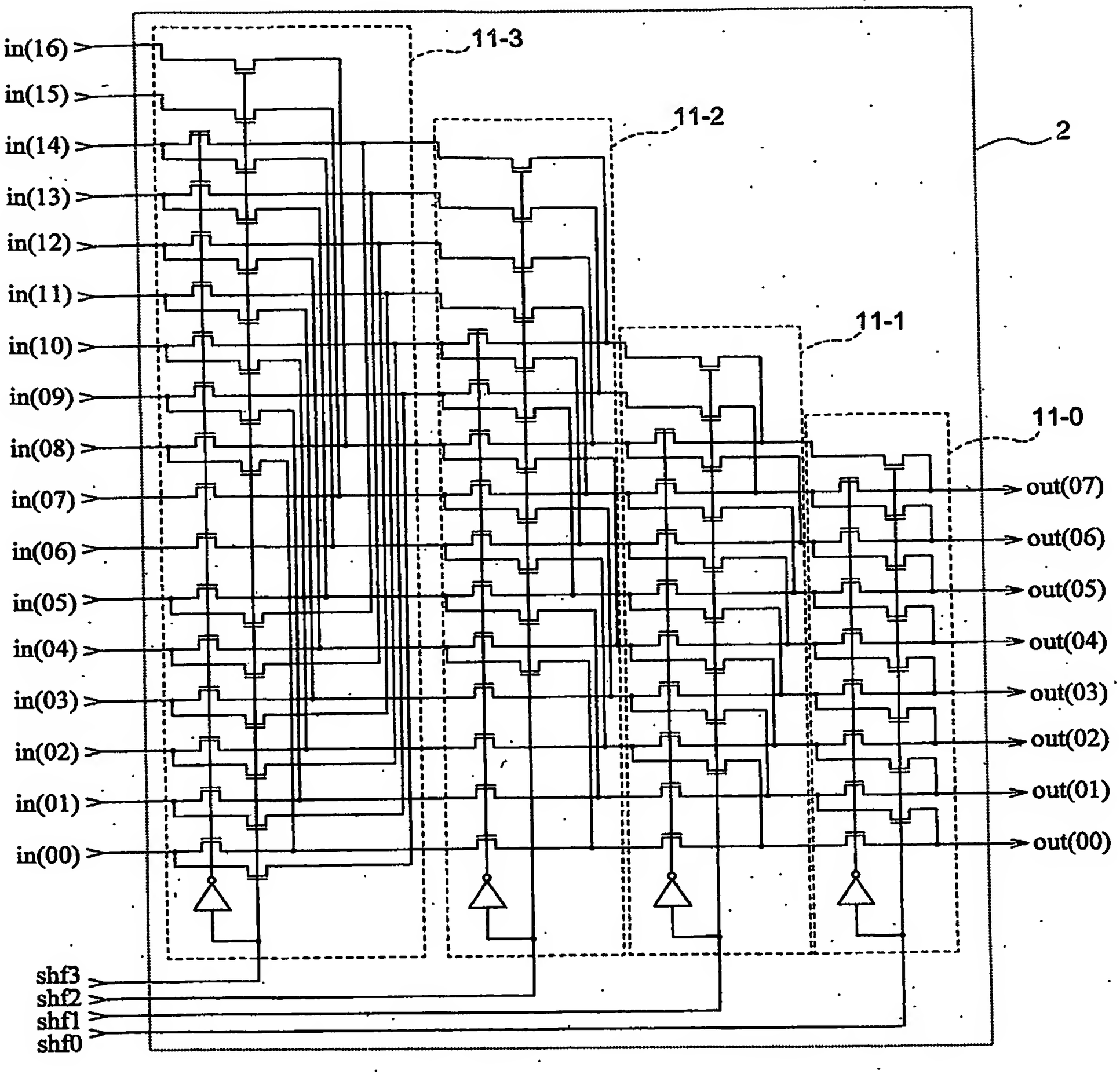
前記マルチプレクサは、前記接続メモリの出力値に従って、前段の前記論理関数メモリの複数の出力線及び複数の前記外部入力線のうちの何れか一本を選択して後段の前記論理関数メモリの入力線に接続するものであること；

を特徴とする請求の範囲第１項乃至第１２項の何れか一記載のプログラマブル論理デバイス。

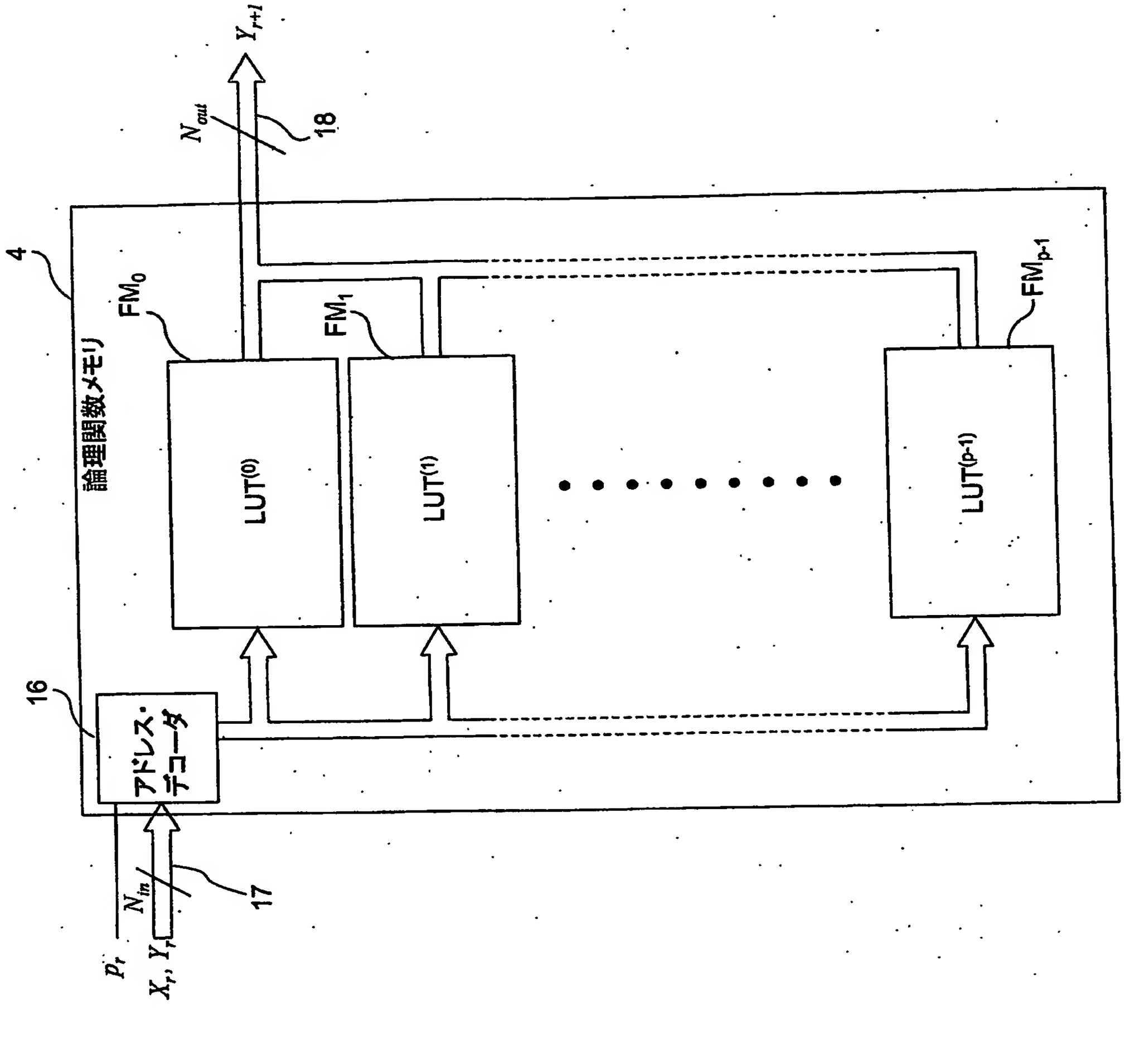
第１図


出力変数Y
7-4
7-3
7-2
7-1
10
1
入力変数X
入力変数レジスタ
演算制御部
3-3
3-2
3-1
3-0
入力選択メモリ
2-3
2-2
2-1
2-0
入力変数選択回路
5-3
5-2
5-1
接続回路
6-3
6-2
6-1
接続メモリ
4-3
4-2
4-1
4-0
論理関数メモリ

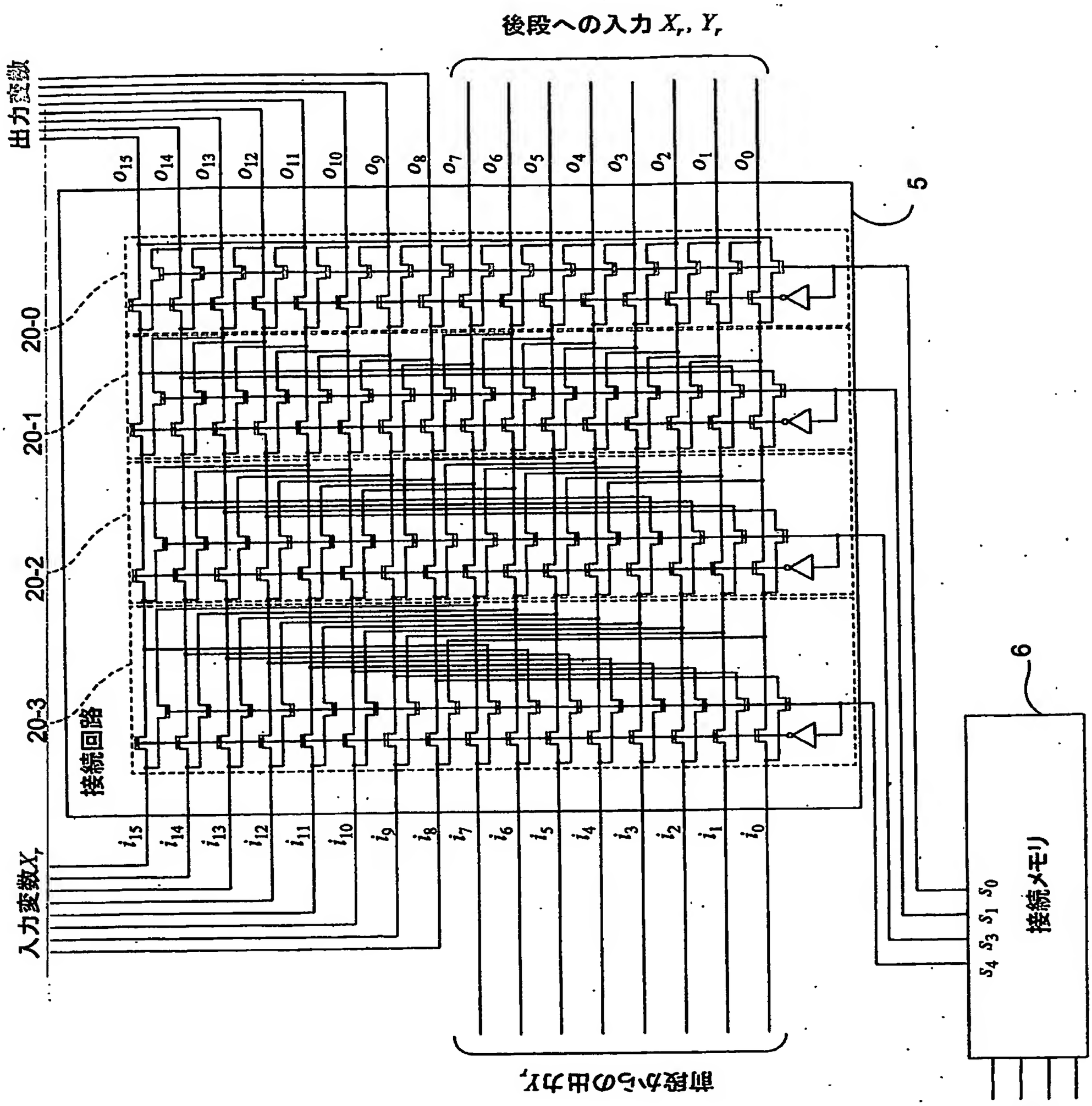
第２図


in(16)
in(15)
in(14)
in(13)
in(12)
in(11)
in(10)
in(09)
in(08)
in(07)
in(06)
in(05)
in(04)
in(03)
in(02)
in(01)
in(00)
11-3
11-2
11-1
11-0
2
out(07)
out(06)
out(05)
out(04)
out(03)
out(02)
out(01)
out(00)
shf3
shf2
shf1
shf0

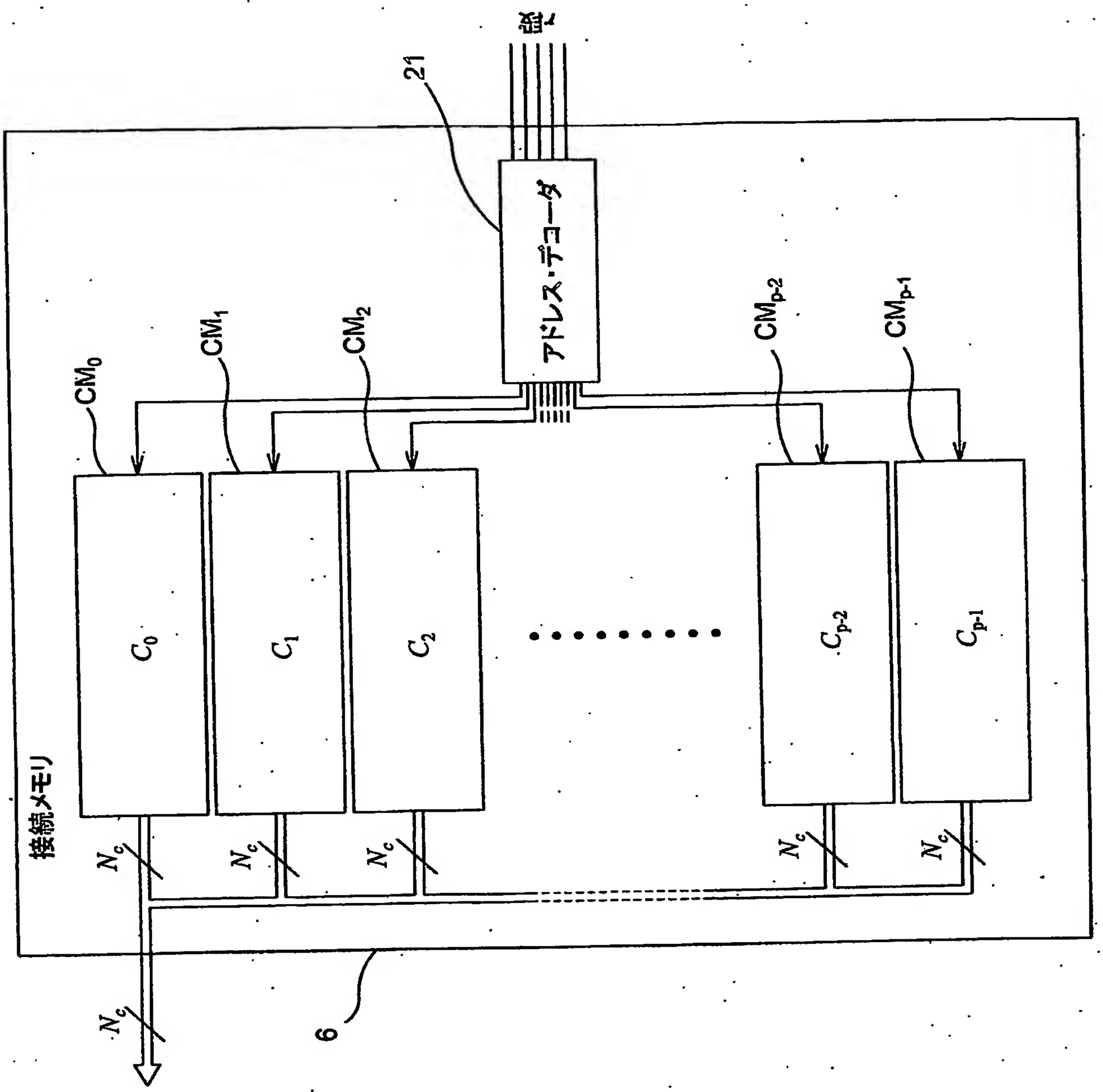
第３図


Y_{r+1}
N_{out}
18
4
論理関数メモリ
FM_0
FM_1
FM_{p-1}
$LUT^{(0)}$
$LUT^{(1)}$
$LUT^{(p-1)}$
16
アドレス・デコーダ
N_{in}
17
p_r
X_r, Y_r

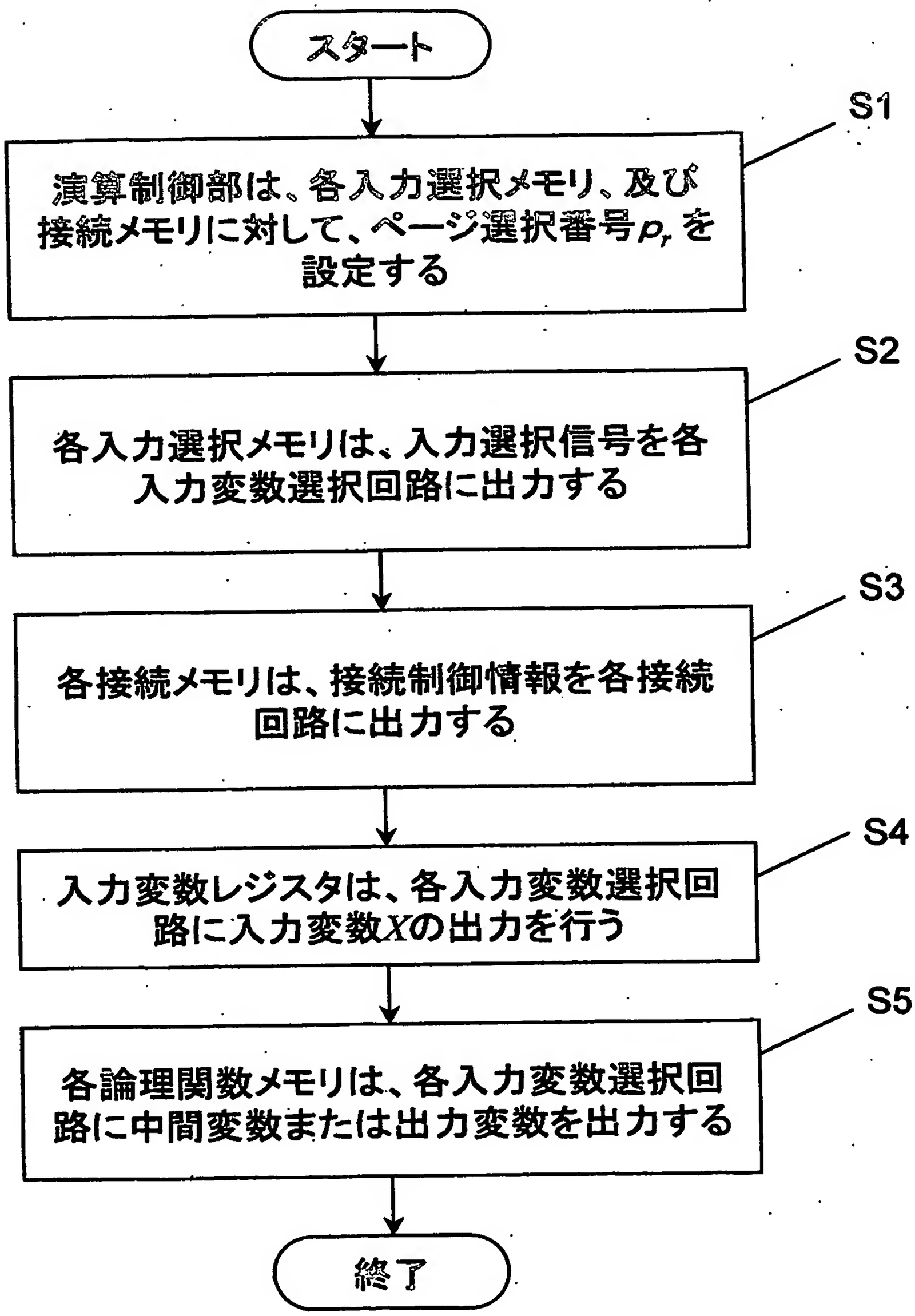
第４図


後段への入力 X_r, Y_r
出力変数
o_{15} o_{14} o_{13} o_{12} o_{11} o_{10} o_9 o_8 o_7 o_6 o_5 o_4 o_3 o_2 o_1 o_0
5
20-0
20-1
20-2
20-3
接続回路
i_{15} i_{14} i_{13} i_{12} i_{11} i_{10} i_9 i_8 i_7 i_6 i_5 i_4 i_3 i_2 i_1 i_0
入力変数 X_r
前段からの出力 Y_r
6
s_4 s_3 s_1 s_0
接続メモリ
r段

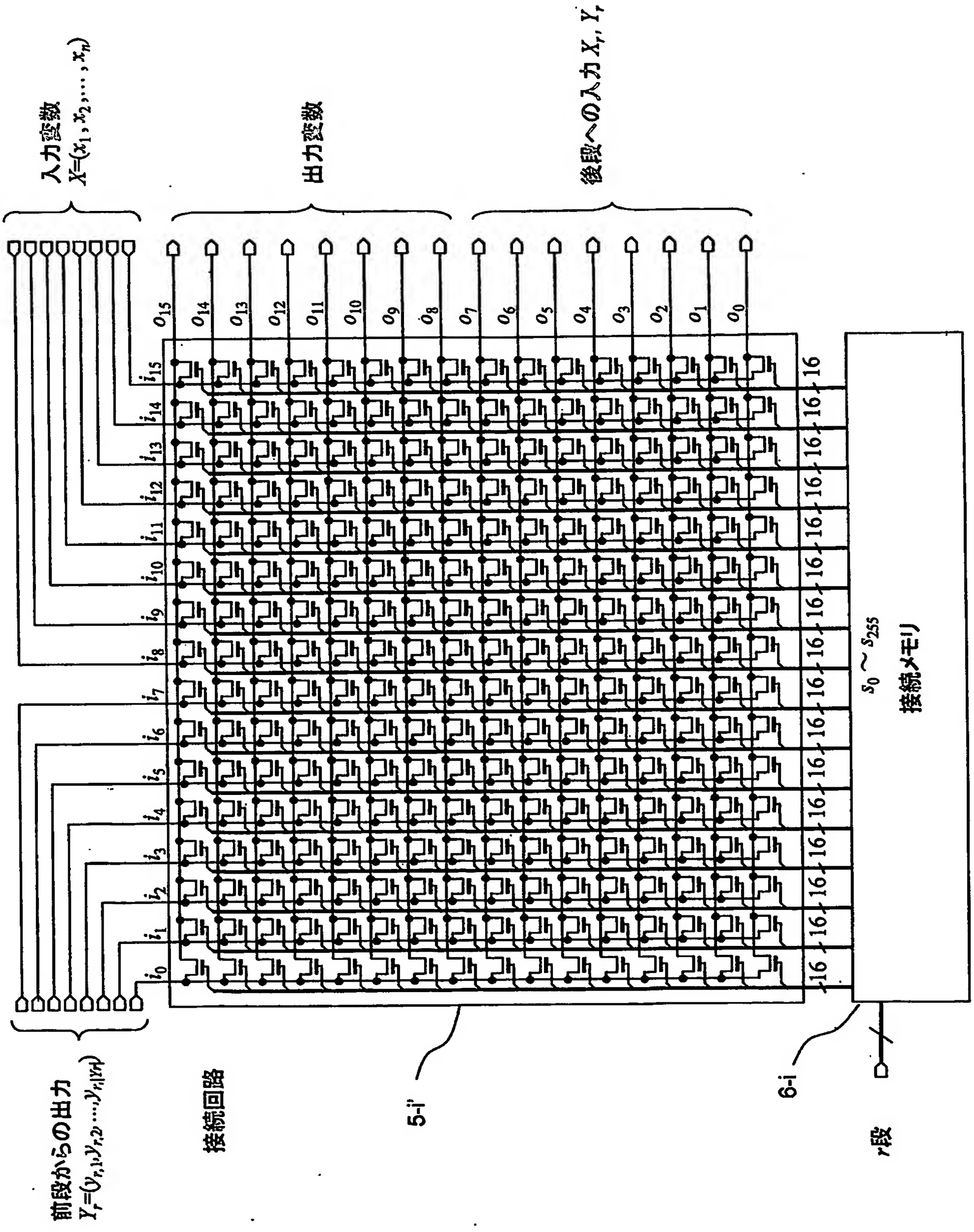
第５図


データ
21
アドレス・デコーダ
CM0
CM1
CM2
CMp-2
CMp-1
C0
C1
C2
Cp-2
Cp-1
接続メモリ
Nc
6

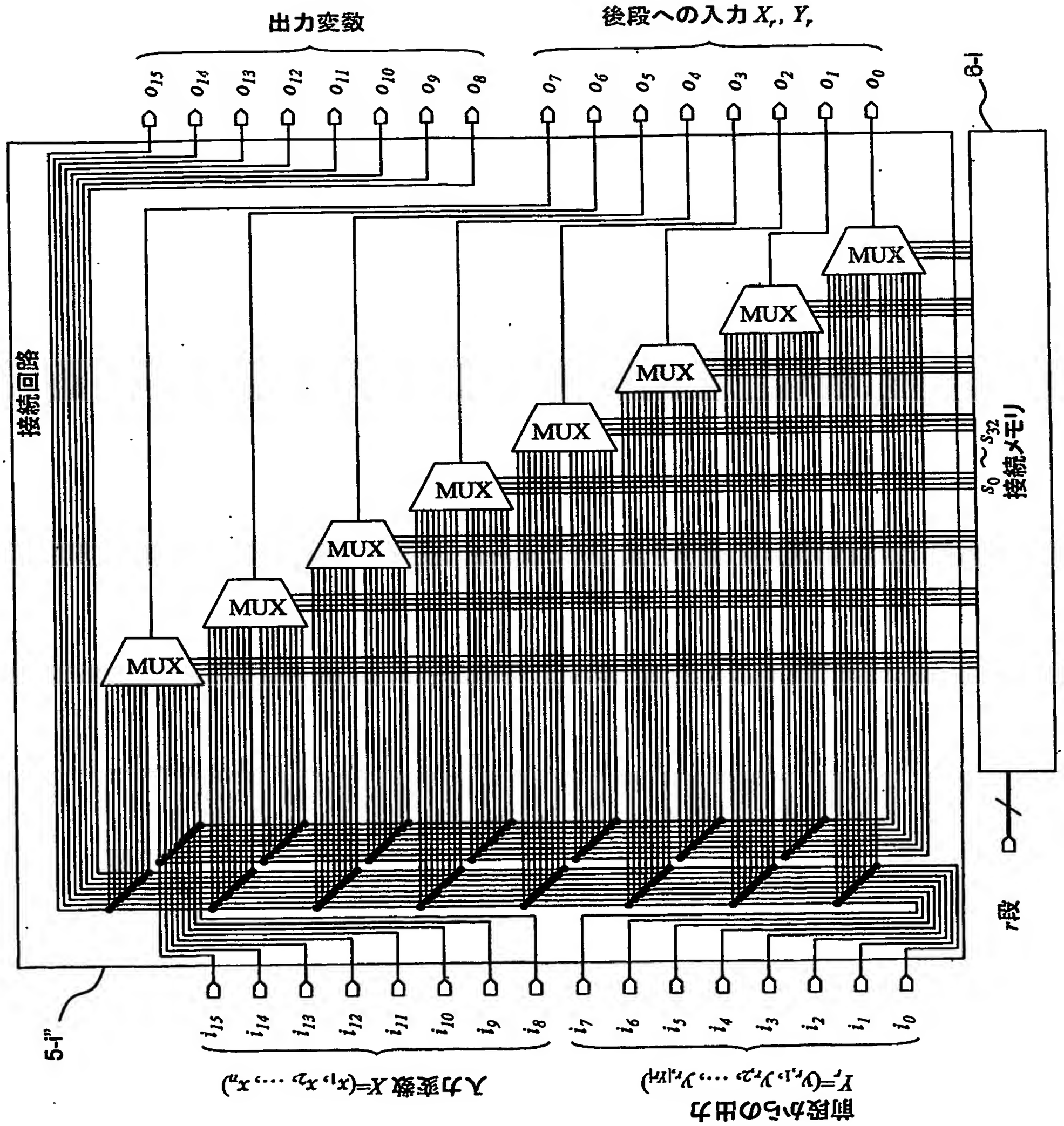
第６図


スタート
S1
演算制御部は、各入力選択メモリ、及び接続メモリに対して、ページ選択番号p_rを設定する
S2
各入力選択メモリは、入力選択信号を各入力変数選択回路に出力する
S3
各接続メモリは、接続制御情報を各接続回路に出力する
S4
入力変数レジスタは、各入力変数選択回路に入力変数Xの出力を行う
S5
各論理関数メモリは、各入力変数選択回路に中間変数または出力変数を出力する
終了

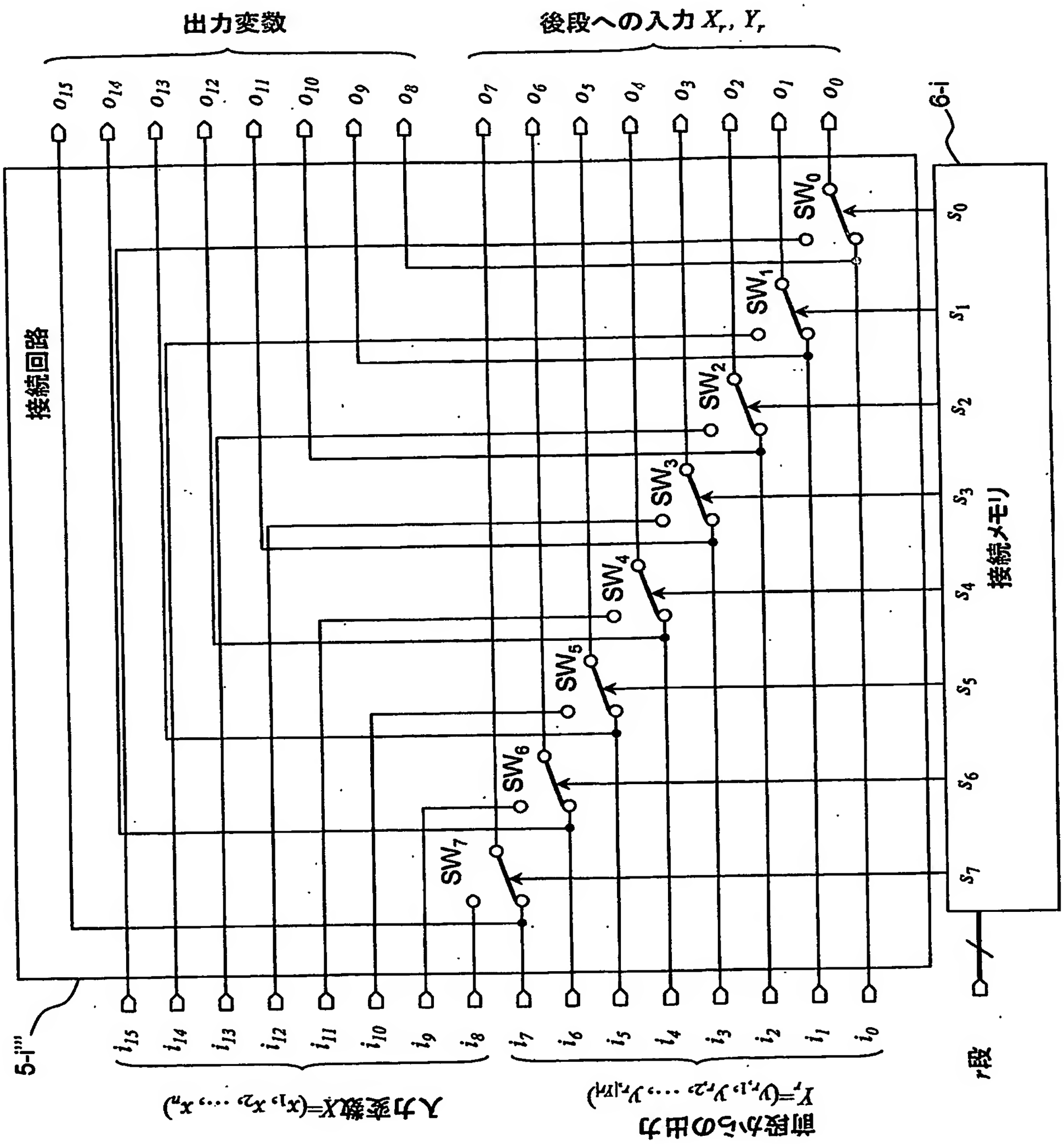
第７図


入力変数
$X=(x_1, x_2, \ldots, x_n)$
出力変数
後段への入力 X_r, Y_r
o_{15} o_{14} o_{13} o_{12} o_{11} o_{10} o_9 o_8 o_7 o_6 o_5 o_4 o_3 o_2 o_1 o_0
i_{15} i_{14} i_{13} i_{12} i_{11} i_{10} i_9 i_8 i_7 i_6 i_5 i_4 i_3 i_2 i_1 i_0
16
$s_0 \sim s_{255}$
接続メモリ
前段からの出力
$Y_r=(y_{r,1}, y_{r,2}, \ldots, y_{r,|Yr|})$
接続回路
5-i'
6-i
r段

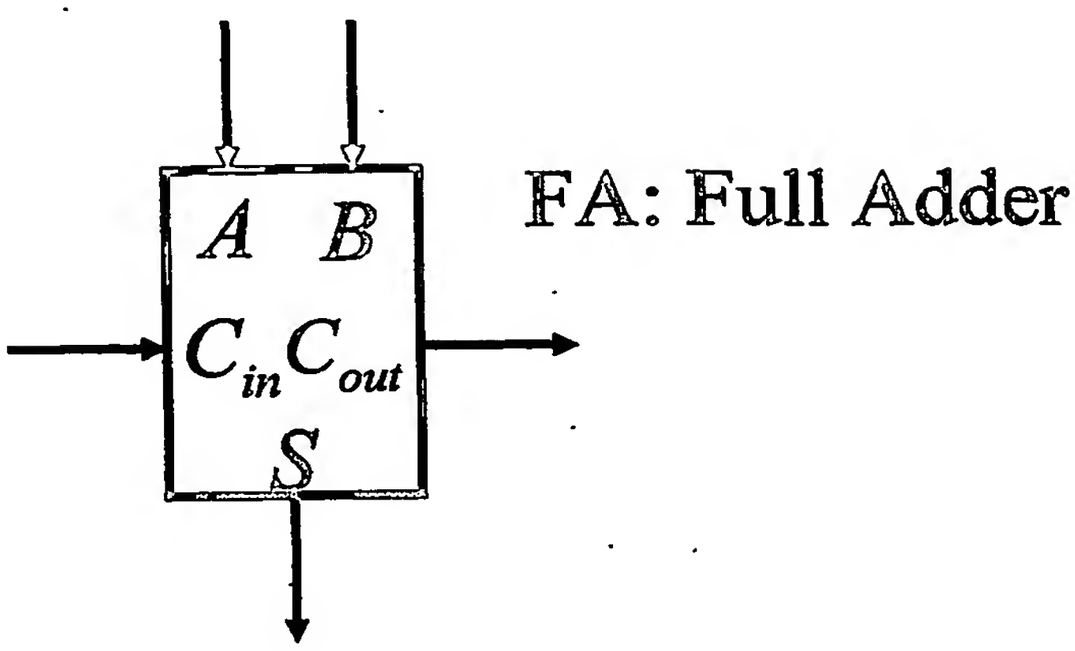
第８図


出力変数
後段への入力 X_r, Y_r
o_{15} o_{14} o_{13} o_{12} o_{11} o_{10} o_9 o_8 o_7 o_6 o_5 o_4 o_3 o_2 o_1 o_0
B-i
接続回路
MUX
$S_0 \sim S_{32}$ 接続メモリ
r段
D
5-i
i_{15} i_{14} i_{13} i_{12} i_{11} i_{10} i_9 i_8 i_7 i_6 i_5 i_4 i_3 i_2 i_1 i_0
入力変数 $X=(x_1, x_2, \cdots, x_n)$
前段からの出力
$Y_r=(y_{r,1}, y_{r,2}, \cdots, y_{r,|Yr|})$

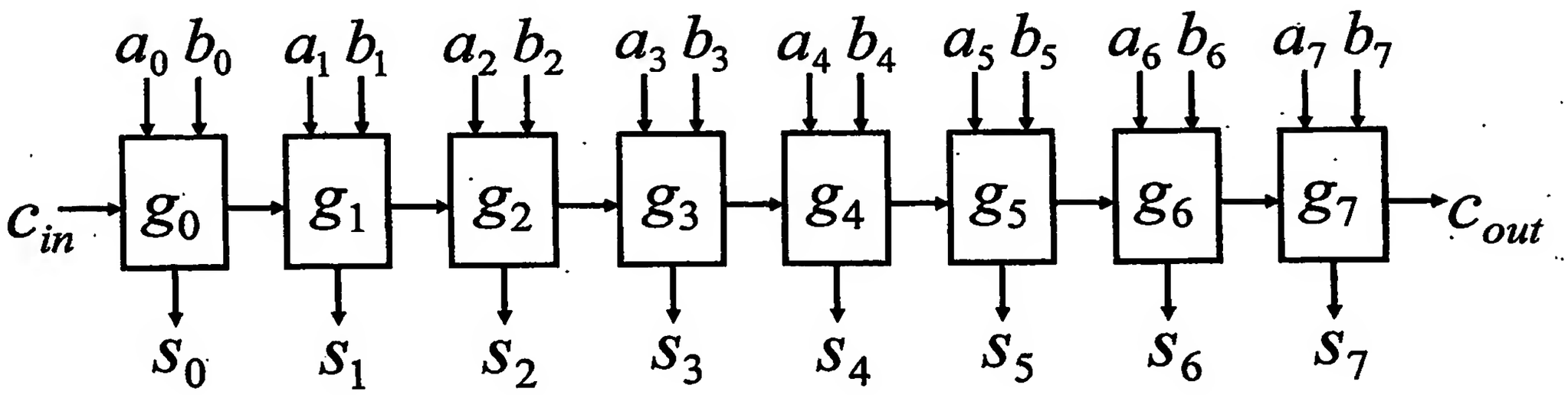
第９図


出力変数
後段への入力 X_r, Y_r
o_{15} o_{14} o_{13} o_{12} o_{11} o_{10} o_9 o_8 o_7 o_6 o_5 o_4 o_3 o_2 o_1 o_0
6-i
接続回路
SW_0 SW_1 SW_2 SW_3 SW_4 SW_5 SW_6 SW_7
s_0 s_1 s_2 s_3 s_4 s_5 s_6 s_7
接続メモリ
5-i'''
i_{15} i_{14} i_{13} i_{12} i_{11} i_{10} i_9 i_8 i_7 i_6 i_5 i_4 i_3 i_2 i_1 i_0
入力変数$X=(x_1, x_2, \cdots, x_n)$
前段からの出力
$Y_r=(y_{r,1}, y_{r,2}, \cdots, y_{r,|Yr|})$
段r

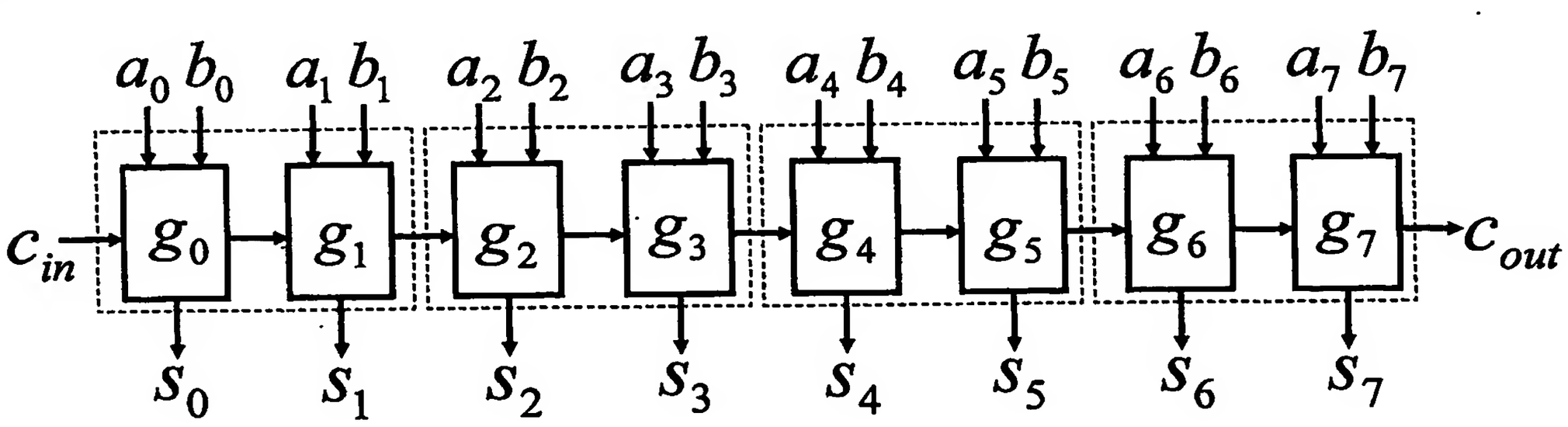
第１０図



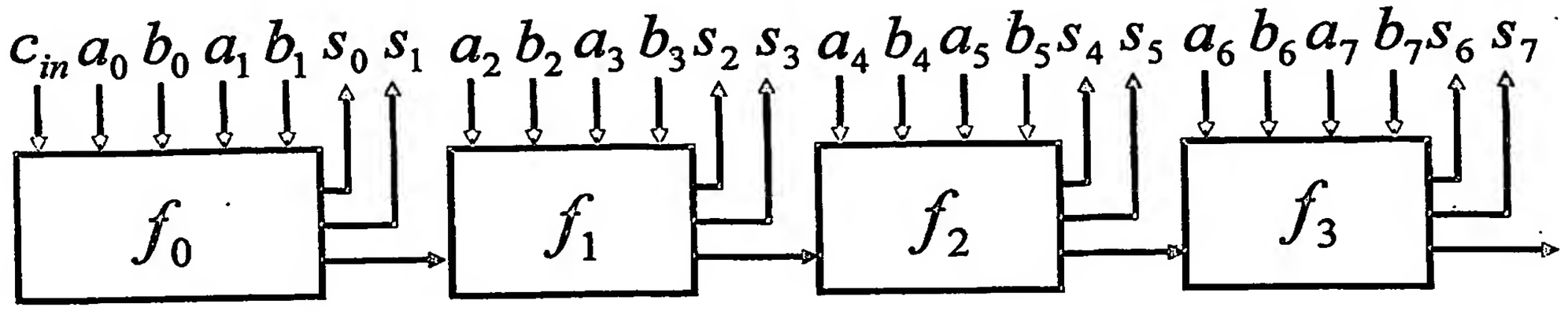
第１１図



第１２図



第１３図


c_{in} a_0 b_0 a_1 b_1 s_0 s_1 a_2 b_2 a_3 b_3 s_2 s_3 a_4 b_4 a_5 b_5 s_4 s_5 a_6 b_6 a_7 b_7 s_6 s_7
f_0
f_1
f_2
f_3

第１４図

c_{in}	a_1	a_0	b_1	b_0	c_{out}	s_1	s_0
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	1	0
0	0	0	1	1	0	1	1
0	0	1	0	0	0	1	1
0	0	1	0	1	0	1	0
0	0	1	1	0	0	0	1
0	0	1	1	1	1	0	0
0	1	0	0	0	0	1	1
0	1	0	0	1	0	1	0
0	1	0	1	0	0	0	1
0	1	0	1	1	1	0	0
0	1	1	0	0	0	0	1
0	1	1	0	1	1	0	0
0	1	1	1	0	1	0	0
0	1	1	1	1	1	0	0
1	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0
1	0	0	1	0	0	0	1
1	0	0	1	1	1	0	0
1	0	1	0	0	0	0	1
1	0	1	0	1	1	0	0
1	0	1	1	0	1	0	0
1	0	1	1	1	1	0	0
1	1	0	0	0	0	0	1
1	1	0	0	1	1	0	0
1	1	0	1	0	1	0	0
1	1	0	1	1	1	0	0
1	1	1	0	0	1	0	0
1	1	1	0	1	1	0	0
1	1	1	1	0	1	0	0
1	1	1	1	1	1	0	0

第１５図

Address								Memory value							
i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7	o_7	o_6	o_5	o_4	o_3	o_2	o_1	o_0
0	0	0	0	0	-	-	-	-	-	-	0	0	0	-	-
0	0	0	0	1	-	-	-	-	-	-	0	0	1	-	-
0	0	0	1	0	-	-	-	-	-	-	0	1	0	-	-
0	0	0	1	1	-	-	-	-	-	-	0	1	1	-	-
0	0	1	0	0	-	-	-	-	-	-	0	0	1	-	-
0	0	1	0	1	-	-	-	-	-	-	0	1	0	-	-
0	0	1	1	0	-	-	-	-	-	-	0	1	1	-	-
0	0	1	1	1	-	-	-	-	-	-	1	0	0	-	-
0	1	0	0	0	-	-	-	-	-	-	0	1	0	-	-
0	1	0	0	1	-	-	-	-	-	-	0	1	1	-	-
0	1	0	1	0	-	-	-	-	-	-	1	0	0	-	-
0	1	0	1	1	-	-	-	-	-	-	1	0	1	-	-
0	1	1	0	0	-	-	-	-	-	-	0	1	1	-	-
0	1	1	0	1	-	-	-	-	-	-	1	0	0	-	-
0	1	1	1	0	-	-	-	-	-	-	1	0	1	-	-
0	1	1	1	1	-	-	-	-	-	-	1	1	0	-	-

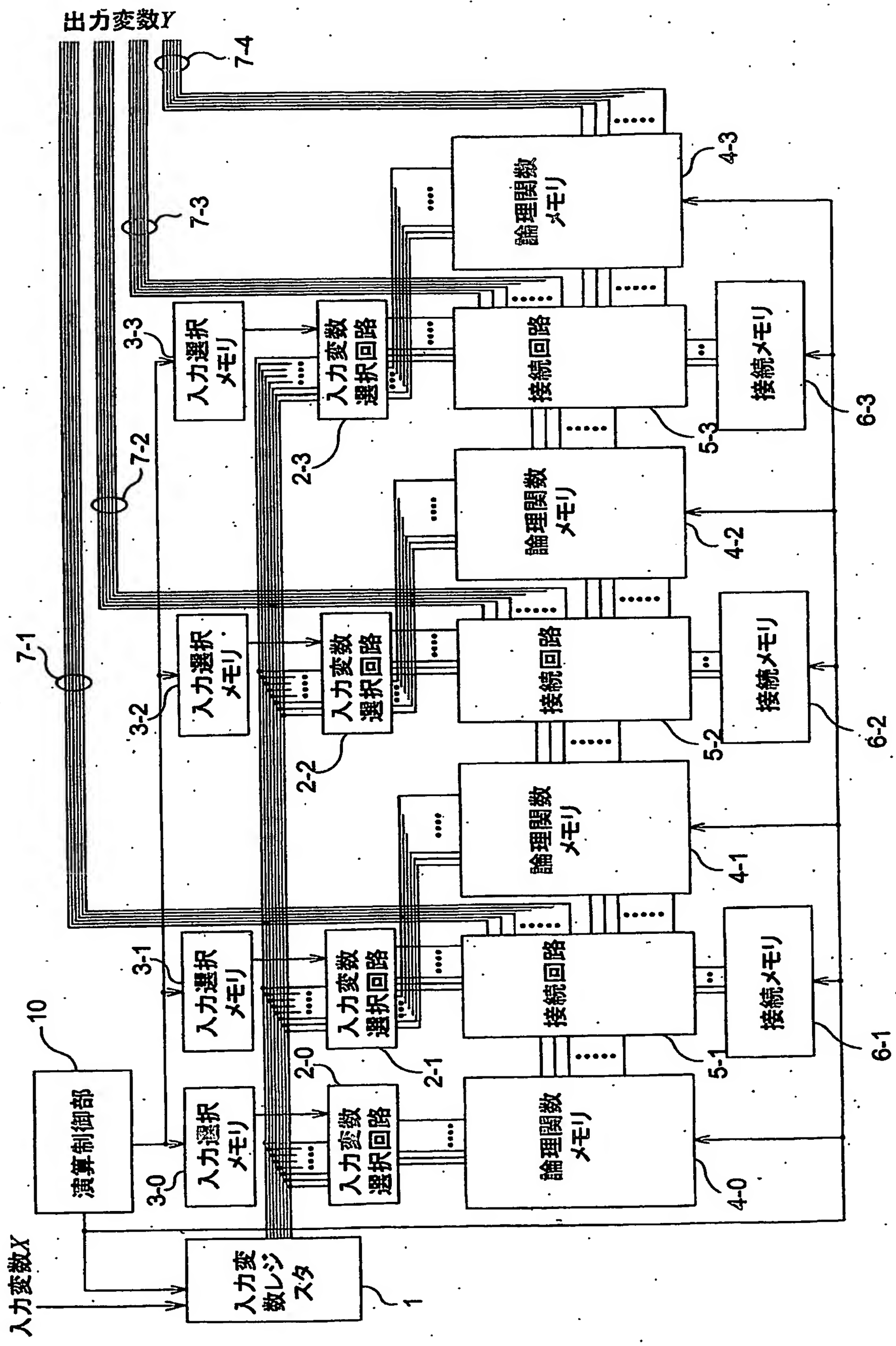
Address								Memory value							
i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7	o_7	o_6	o_5	o_4	o_3	o_2	o_1	o_0
1	0	0	0	0	-	-	-	-	-	-	0	0	1	-	-
1	0	0	0	1	-	-	-	-	-	-	0	1	0	-	-
1	0	0	1	0	-	-	-	-	-	-	0	1	1	-	-
1	0	0	1	1	-	-	-	-	-	-	1	0	0	-	-
1	0	1	0	0	-	-	-	-	-	-	0	1	0	-	-
1	0	1	0	1	-	-	-	-	-	-	0	1	1	-	-
1	0	1	1	0	-	-	-	-	-	-	1	0	0	-	-
1	0	1	1	1	-	-	-	-	-	-	1	0	1	-	-
1	1	0	0	0	-	-	-	-	-	-	0	1	1	-	-
1	1	0	0	1	-	-	-	-	-	-	1	0	0	-	-
1	1	0	1	0	-	-	-	-	-	-	1	0	1	-	-
1	1	0	1	1	-	-	-	-	-	-	1	1	0	-	-
1	1	1	0	0	-	-	-	-	-	-	1	0	0	-	-
1	1	1	0	1	-	-	-	-	-	-	1	0	1	-	-
1	1	1	1	0	-	-	-	-	-	-	1	1	0	-	-
1	1	1	1	1	-	-	-	-	-	-	1	1	1	-	-

第１６図

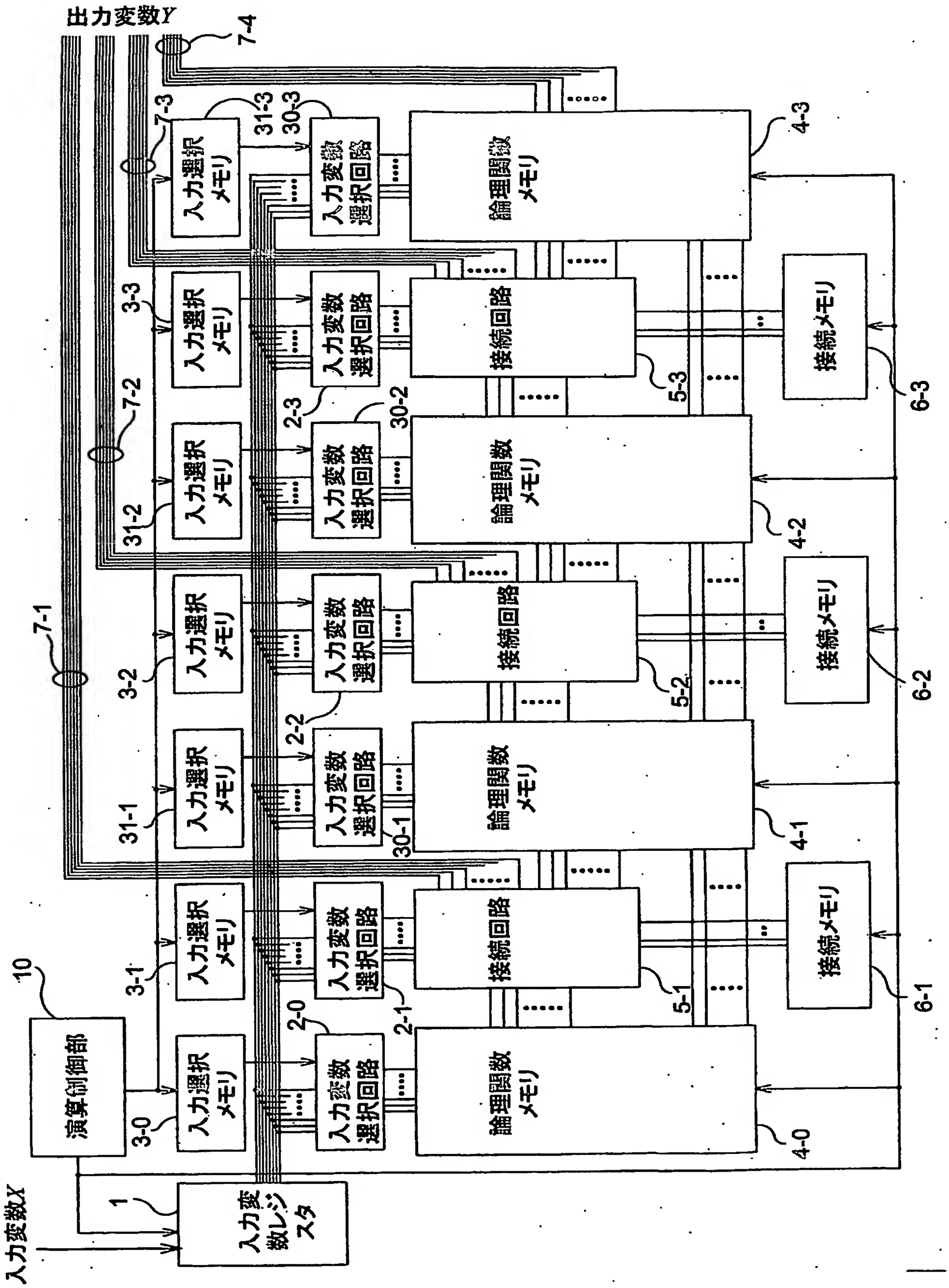
Address								Memory value							
i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7	o_7	o_6	o_5	o_4	o_3	o_2	o_1	o_0
0	-	-	-	0	0	0	0	-	-	-	0	0	0	-	-
0	-	-	-	0	0	0	1	-	-	-	0	0	1	-	-
0	-	-	-	0	0	1	0	-	-	-	0	1	0	-	-
0	-	-	-	0	0	1	1	-	-	-	0	1	1	-	-
0	-	-	-	0	1	0	0	-	-	-	0	0	1	-	-
0	-	-	-	0	1	0	1	-	-	-	0	1	0	-	-
0	-	-	-	0	1	1	0	-	-	-	0	1	1	-	-
0	-	-	-	0	1	1	1	-	-	-	1	0	0	-	-
0	-	-	-	1	0	0	0	-	-	-	0	1	0	-	-
0	-	-	-	1	0	0	1	-	-	-	0	1	1	-	-
0	-	-	-	1	0	1	0	-	-	-	1	0	0	-	-
0	-	-	-	1	0	1	1	-	-	-	1	0	1	-	-
0	-	-	-	1	1	0	0	-	-	-	0	1	1	-	-
0	-	-	-	1	1	0	1	-	-	-	1	0	0	-	-
0	-	-	-	1	1	1	0	-	-	-	1	0	1	-	-
0	-	-	-	1	1	1	1	-	-	-	1	1	0	-	-

Address								Memory value							
i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7	o_7	o_6	o_5	o_4	o_3	o_2	o_1	o_0
1	-	-	-	0	0	0	0	-	-	-	0	0	1	-	-
1	-	-	-	0	0	0	1	-	-	-	0	1	0	-	-
1	-	-	-	0	0	1	0	-	-	-	0	1	1	-	-
1	-	-	-	0	0	1	1	-	-	-	1	0	0	-	-
1	-	-	-	0	1	0	0	-	-	-	0	1	0	-	-
1	-	-	-	0	1	0	1	-	-	-	0	1	1	-	-
1	-	-	-	0	1	1	0	-	-	-	1	0	0	-	-
1	-	-	-	0	1	1	1	-	-	-	1	0	1	-	-
1	-	-	-	1	0	0	0	-	-	-	0	1	1	-	-
1	-	-	-	1	0	0	1	-	-	-	1	0	0	-	-
1	-	-	-	1	0	1	0	-	-	-	1	0	1	-	-
1	-	-	-	1	0	1	1	-	-	-	1	1	0	-	-
1	-	-	-	1	1	0	0	-	-	-	1	0	0	-	-
1	-	-	-	1	1	0	1	-	-	-	1	0	1	-	-
1	-	-	-	1	1	1	0	-	-	-	1	1	0	-	-
1	-	-	-	1	1	1	1	-	-	-	1	1	1	-	-

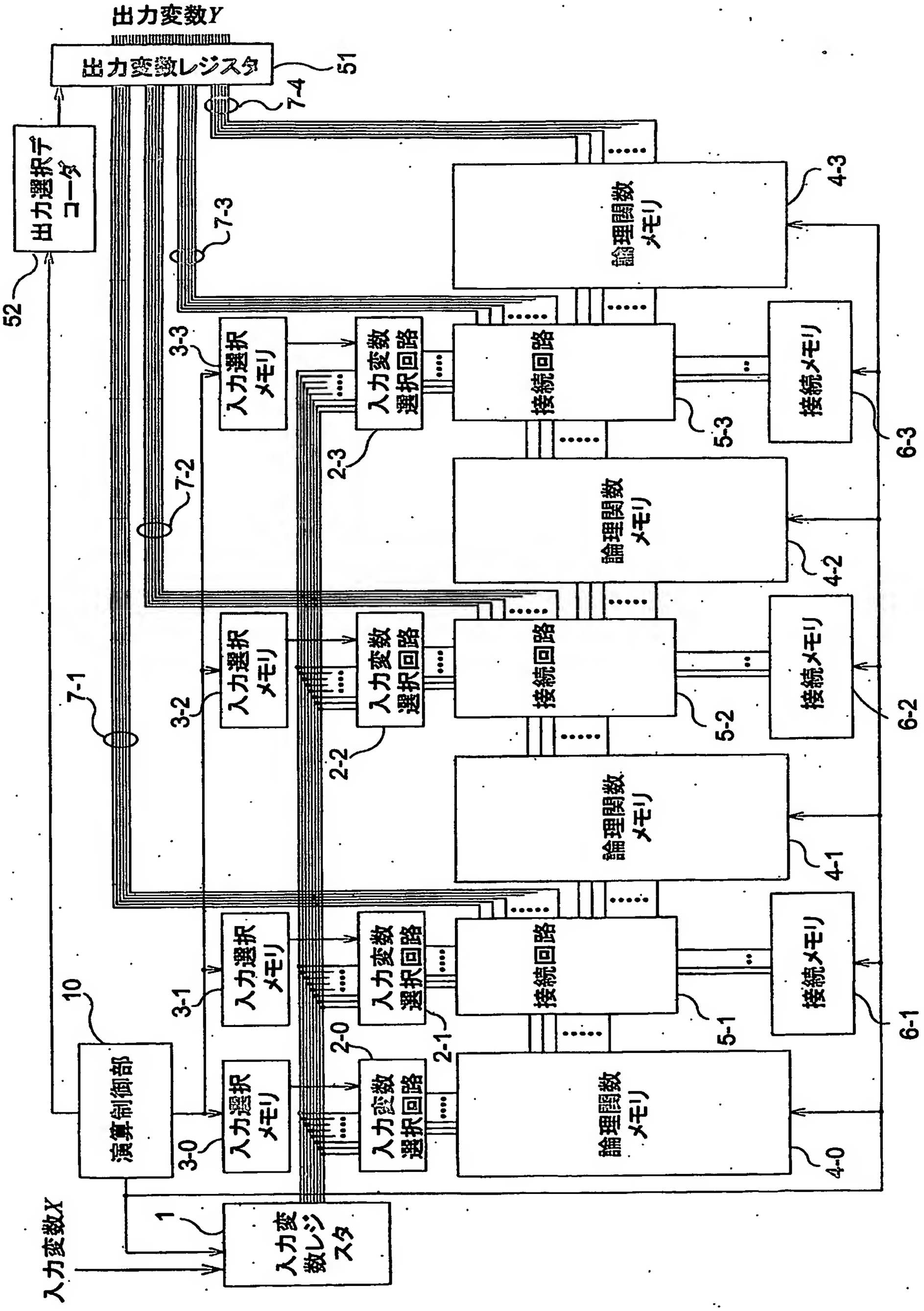
第１７図


出力変数Y
7-4
7-3
7-2
7-1
3-3
3-2
3-1
3-0
入力選択メモリ
2-3
2-2
2-1
2-0
入力変数選択回路
論理関数メモリ
4-3
4-2
4-1
4-0
接続回路
5-3
5-2
5-1
接続メモリ
6-3
6-2
6-1
演算制御部
10
入力変数レジスタ
1
入力変数X

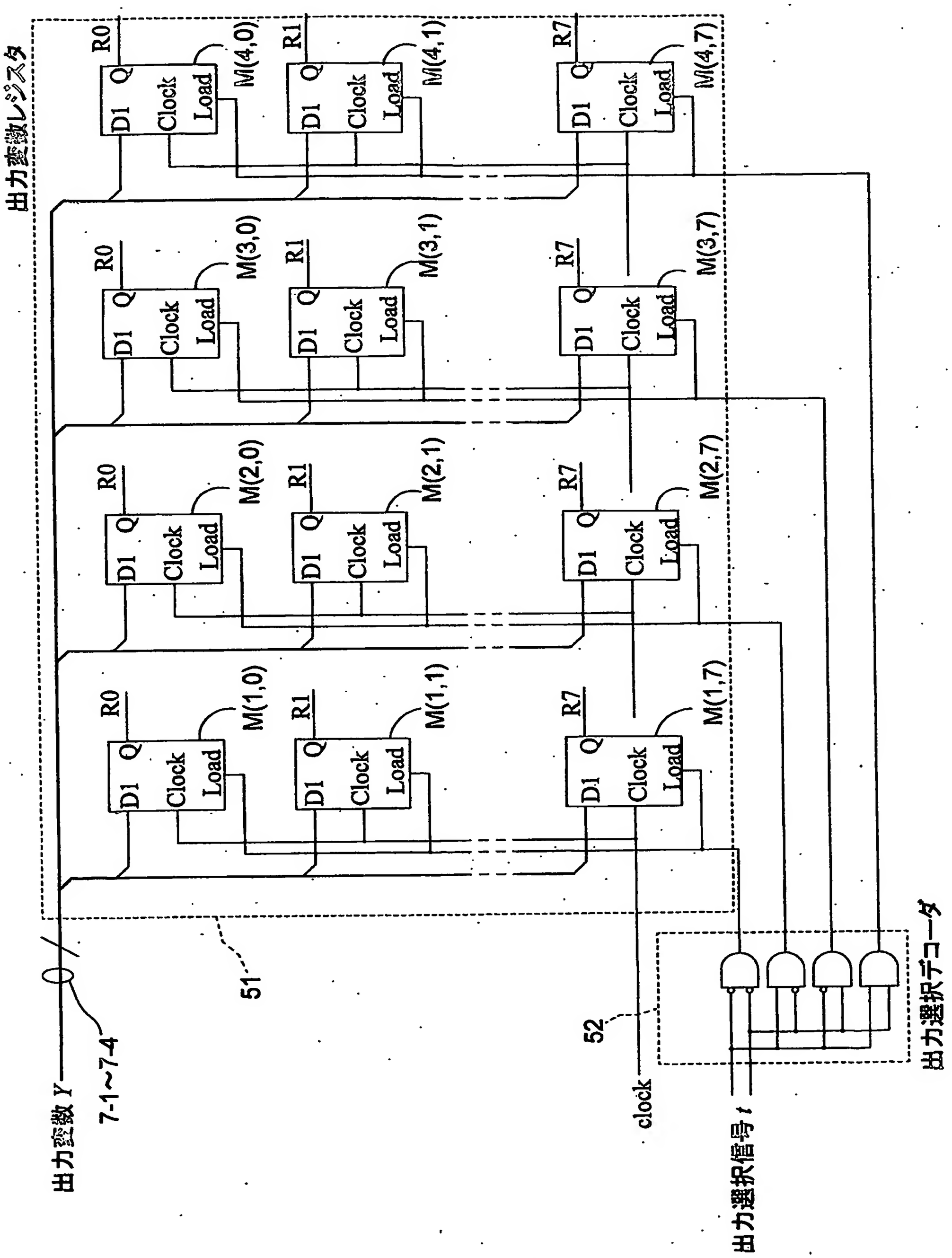
第１８図


出力変数Y
7-4
7-3
31-3
30-3
入力選択メモリ
入力変数選択回路
論理関数メモリ
4-3
3-3
7-2
31-2
2-3
30-2
接続回路
5-3
接続メモリ
6-3
3-2
2-2
5-2
4-2
7-1
31-1
30-1
3-1
2-1
5-1
4-1
6-2
6-1
10
演算制御部
3-0
2-0
4-0
6-0
入力変数X
1
入力変数レジスタ

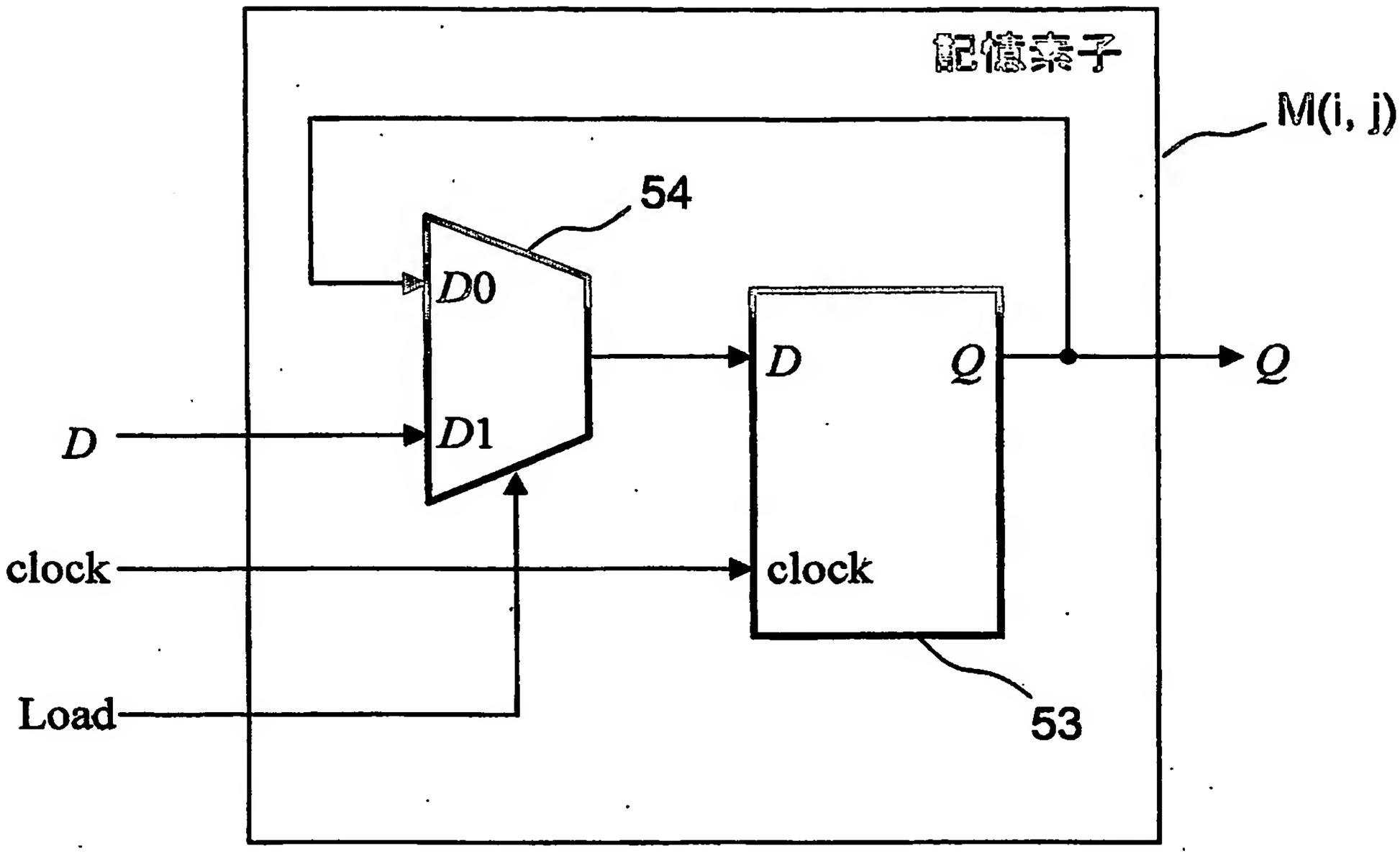
第１９図


出力変数Y
出力変数レジスタ
51
7-4
出力選択デコーダ
52
7-3
論理関数メモリ
4-3
入力選択メモリ
3-3
入力変数選択回路
2-3
接続回路
5-3
接続メモリ
6-3
7-2
論理関数メモリ
4-2
入力選択メモリ
3-2
入力変数選択回路
2-2
接続回路
5-2
接続メモリ
6-2
7-1
論理関数メモリ
4-1
入力選択メモリ
3-1
入力変数選択回路
2-1
接続回路
5-1
接続メモリ
6-1
10
演算制御部
入力選択メモリ
3-0
入力変数選択回路
2-0
論理関数メモリ
4-0
1
入力変数レジスタ
入力変数X

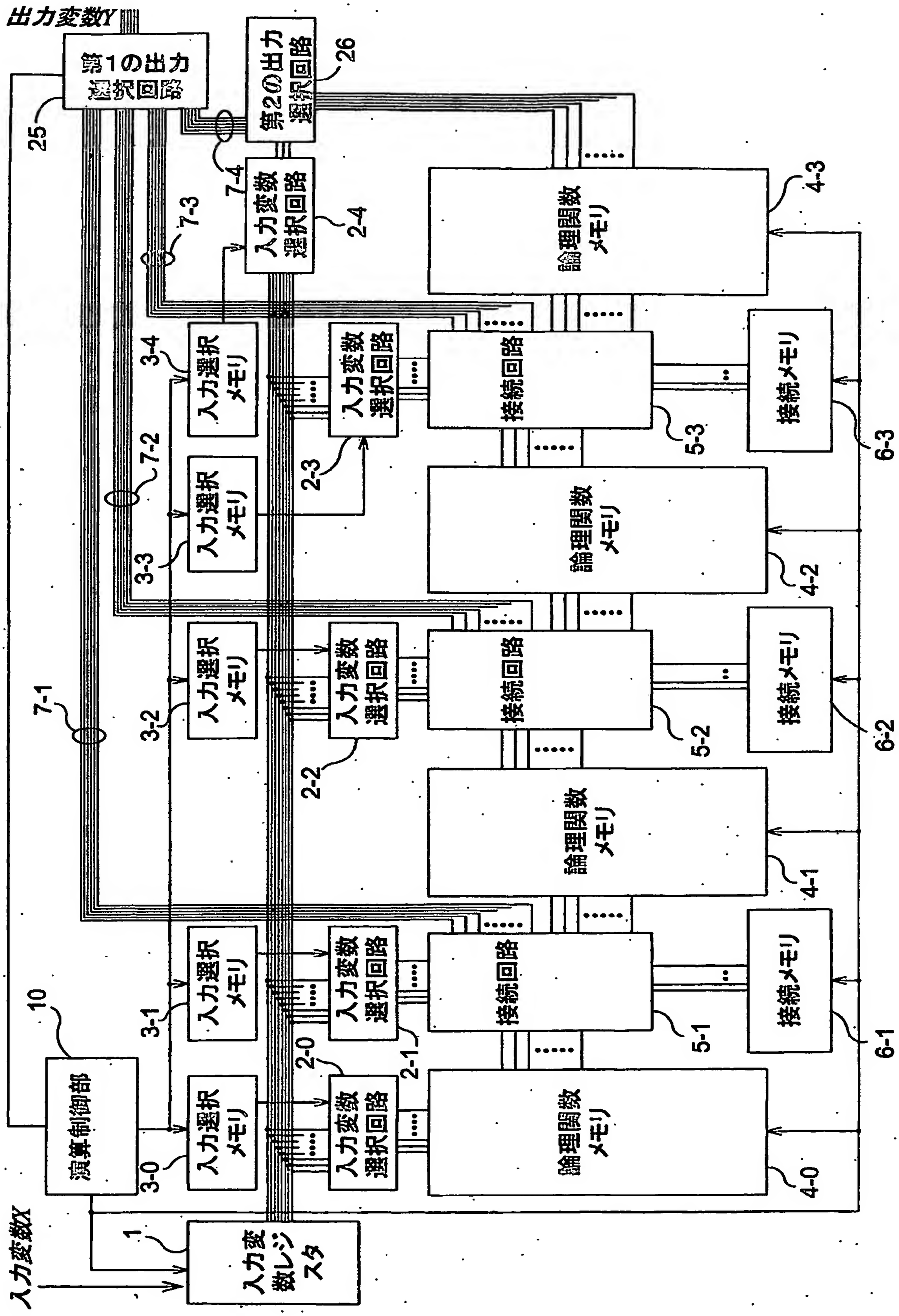
第２０図


出力変数レジスタ
R0
R1
R7
D1
Q
Clock
Load
M(4,0)
M(4,1)
M(4,7)
M(3,0)
M(3,1)
M(3,7)
M(2,0)
M(2,1)
M(2,7)
M(1,0)
M(1,1)
M(1,7)
出力変数 Y
7-1～7-4
51
52
clock
出力選択信号 t
出力選択デコーダ

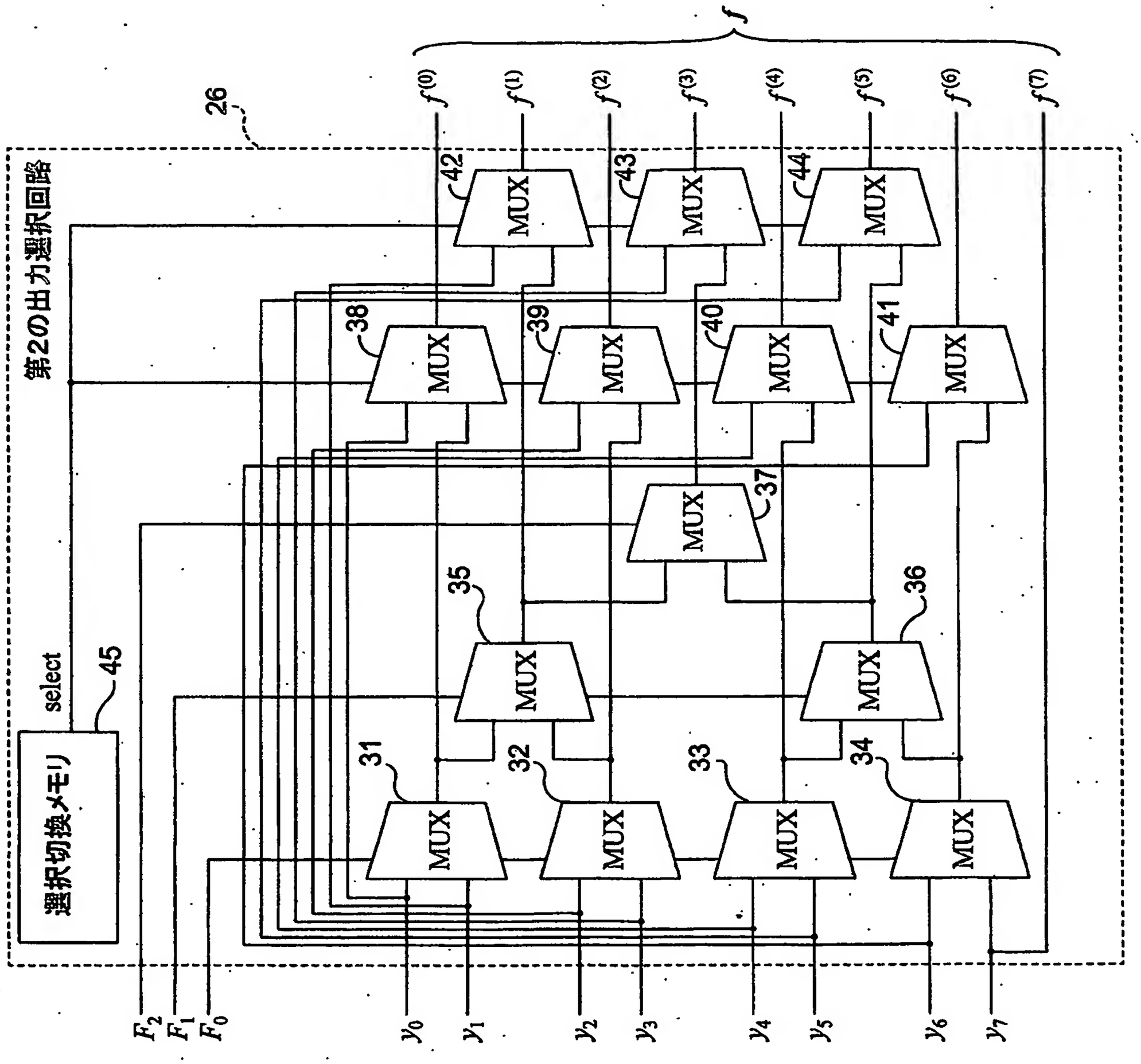
第２１図



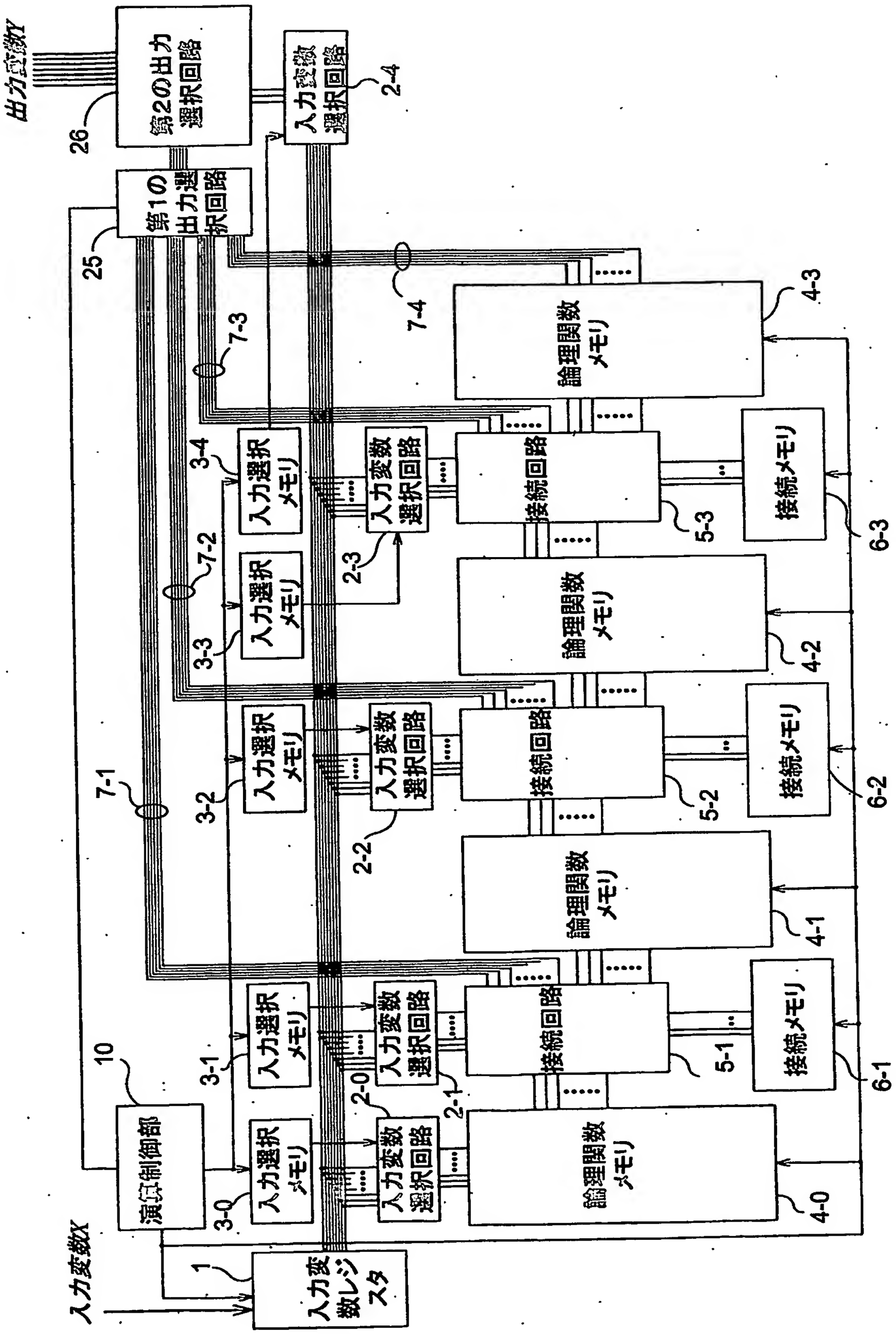
第２２図


出力変数Y
第1の出力選択回路
25
第2の出力選択回路
26
7-4
入力変数選択回路
2-4
7-3
論理関数メモリ
4-3
入力選択メモリ
3-4
入力変数選択回路
2-3
接続回路
5-3
接続メモリ
6-3
7-2
入力選択メモリ
3-3
論理関数メモリ
4-2
入力選択メモリ
3-2
入力変数選択回路
2-2
接続回路
5-2
接続メモリ
6-2
7-1
論理関数メモリ
4-1
入力選択メモリ
3-1
入力変数選択回路
2-1
接続回路
5-1
接続メモリ
6-1
10
演算制御部
3-0
入力選択メモリ
入力変数選択回路
2-0
論理関数メモリ
4-0
入力変数X
1
入力変数レジスタ

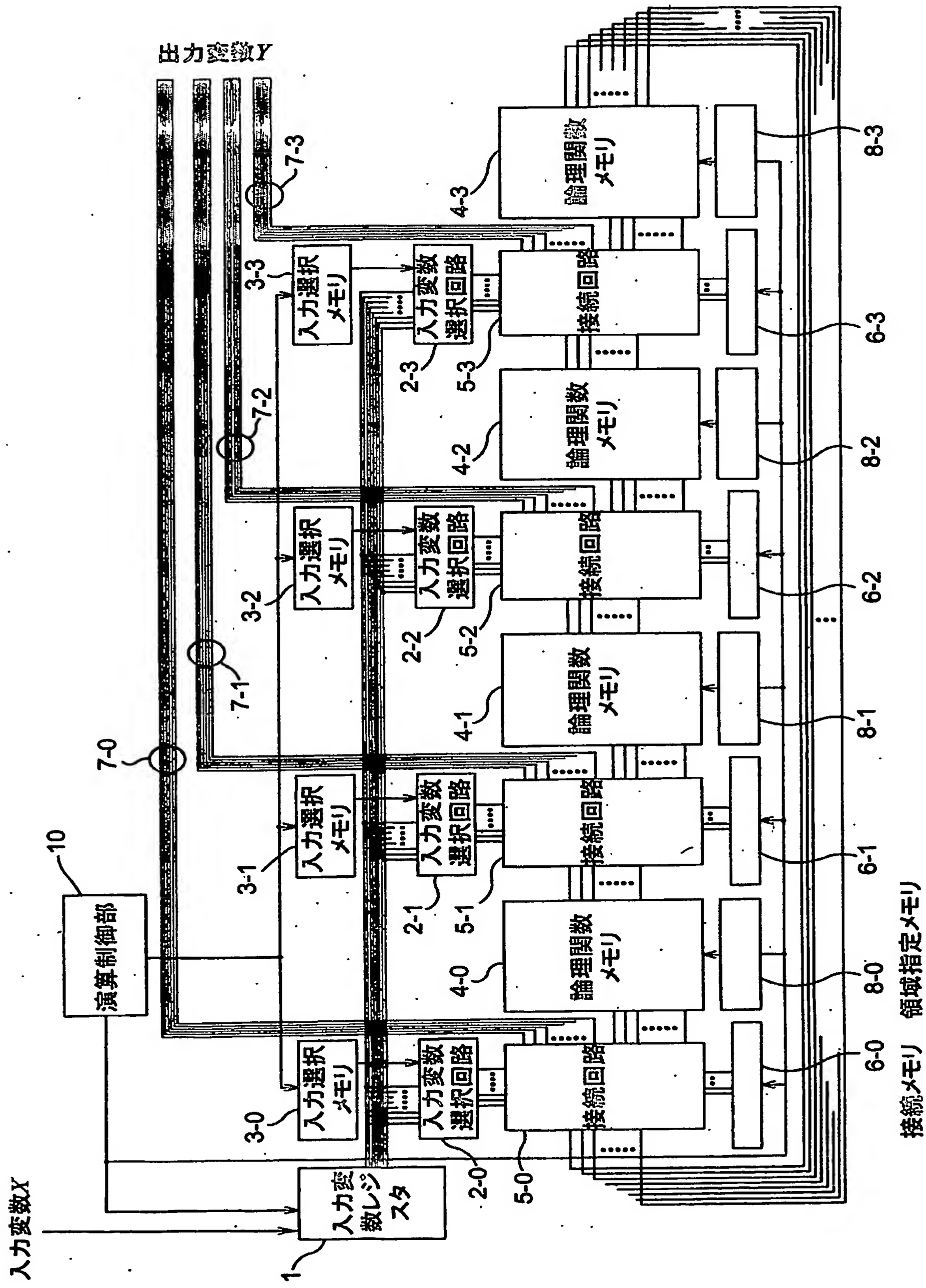
第２３図


f
f(0)
f(1)
f(2)
f(3)
f(4)
f(5)
f(6)
f(7)
26
第2の出力選択回路
選択切換メモリ
select
45
MUX
31
32
33
34
35
36
37
38
39
40
41
42
43
44
F2
F1
F0
y0
y1
y2
y3
y4
y5
y6
y7

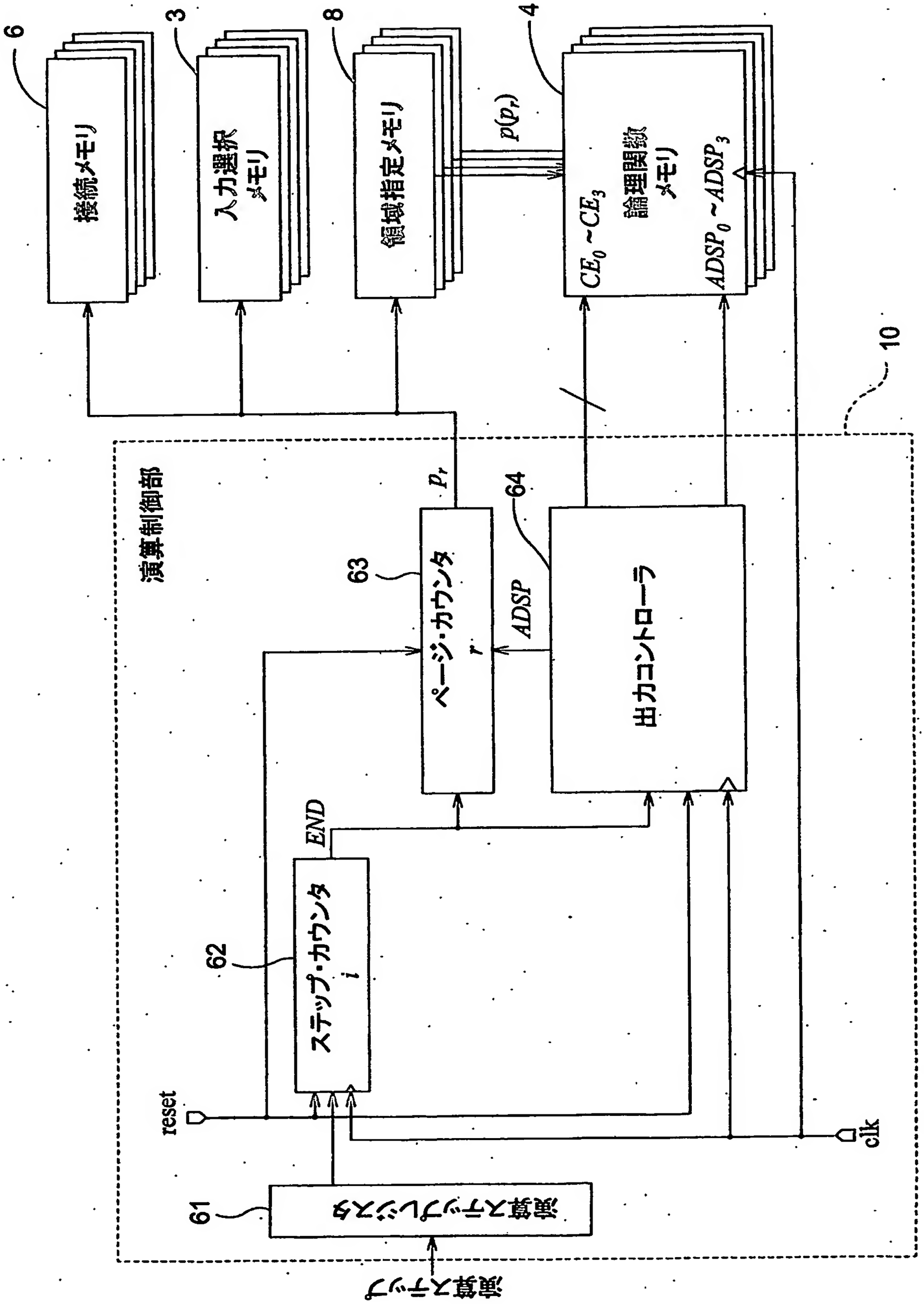
第２４図


出力変数Y
第2の出力選択回路
26
入力変数選択回路
2-4
第1の出力選択回路
25
7-4
7-3
論理関数メモリ
4-3
入力選択メモリ
3-4
入力変数選択回路
接続回路
5-3
接続メモリ
6-3
7-2
入力選択メモリ
3-3
2-3
論理関数メモリ
4-2
入力選択メモリ
3-2
入力変数選択回路
接続回路
5-2
接続メモリ
6-2
7-1
2-2
論理関数メモリ
4-1
入力選択メモリ
3-1
入力変数選択回路
接続回路
5-1
接続メモリ
6-1
10
演算制御部
2-0
2-1
入力選択メモリ
3-0
入力変数選択回路
論理関数メモリ
4-0
入力変数X
1
入力変数レジスタ

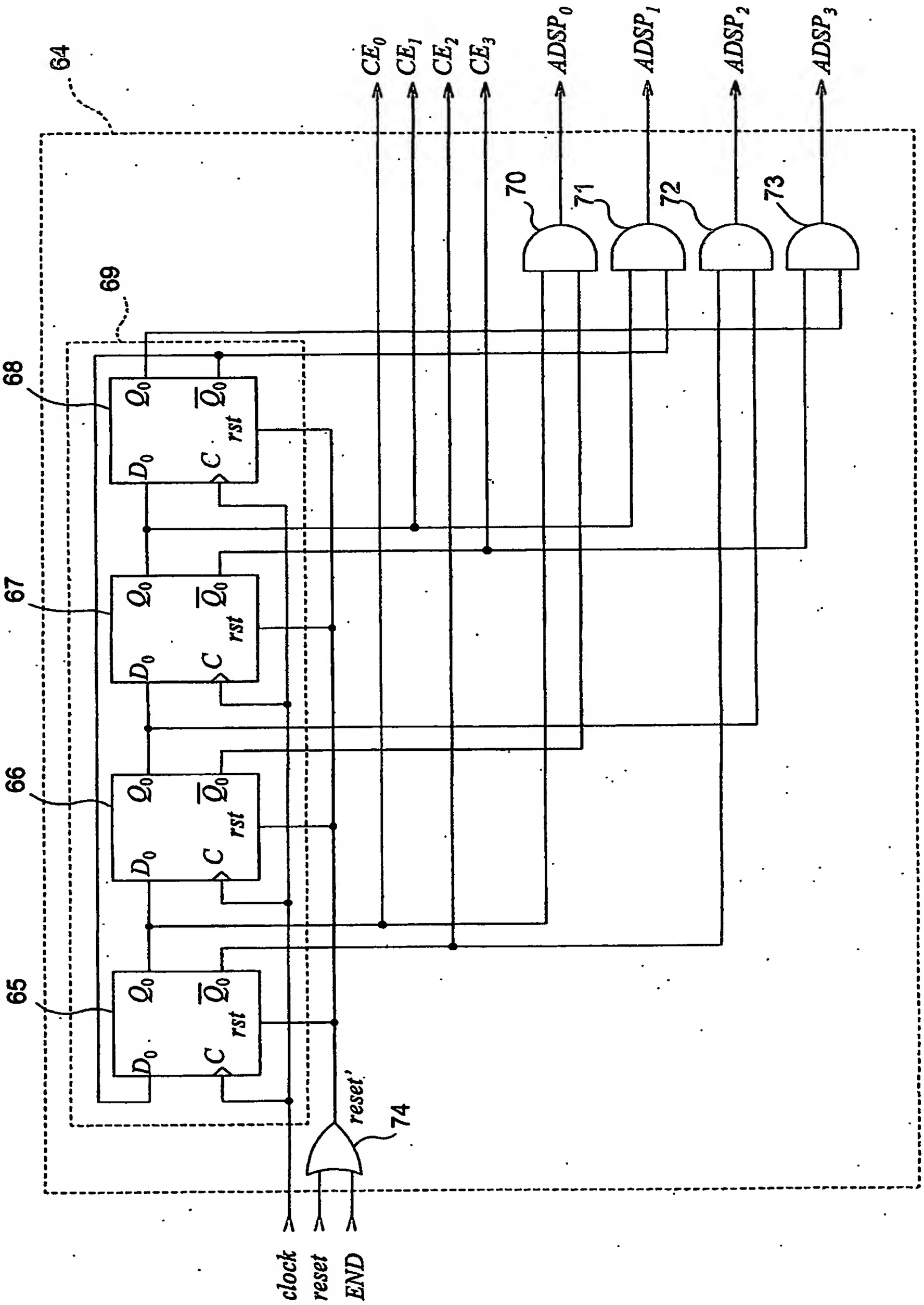
第２５図


出力変数Y
入力変数X
演算制御部
入力変数レジスタ
入力選択メモリ
入力変数選択回路
論理関数メモリ
接続回路
接続メモリ　領域指定メモリ
1
10
2-0
2-1
2-2
2-3
3-0
3-1
3-2
3-3
4-0
4-1
4-2
4-3
5-0
5-1
5-2
5-3
6-0
6-1
6-2
6-3
7-0
7-1
7-2
7-3
8-0
8-1
8-2
8-3

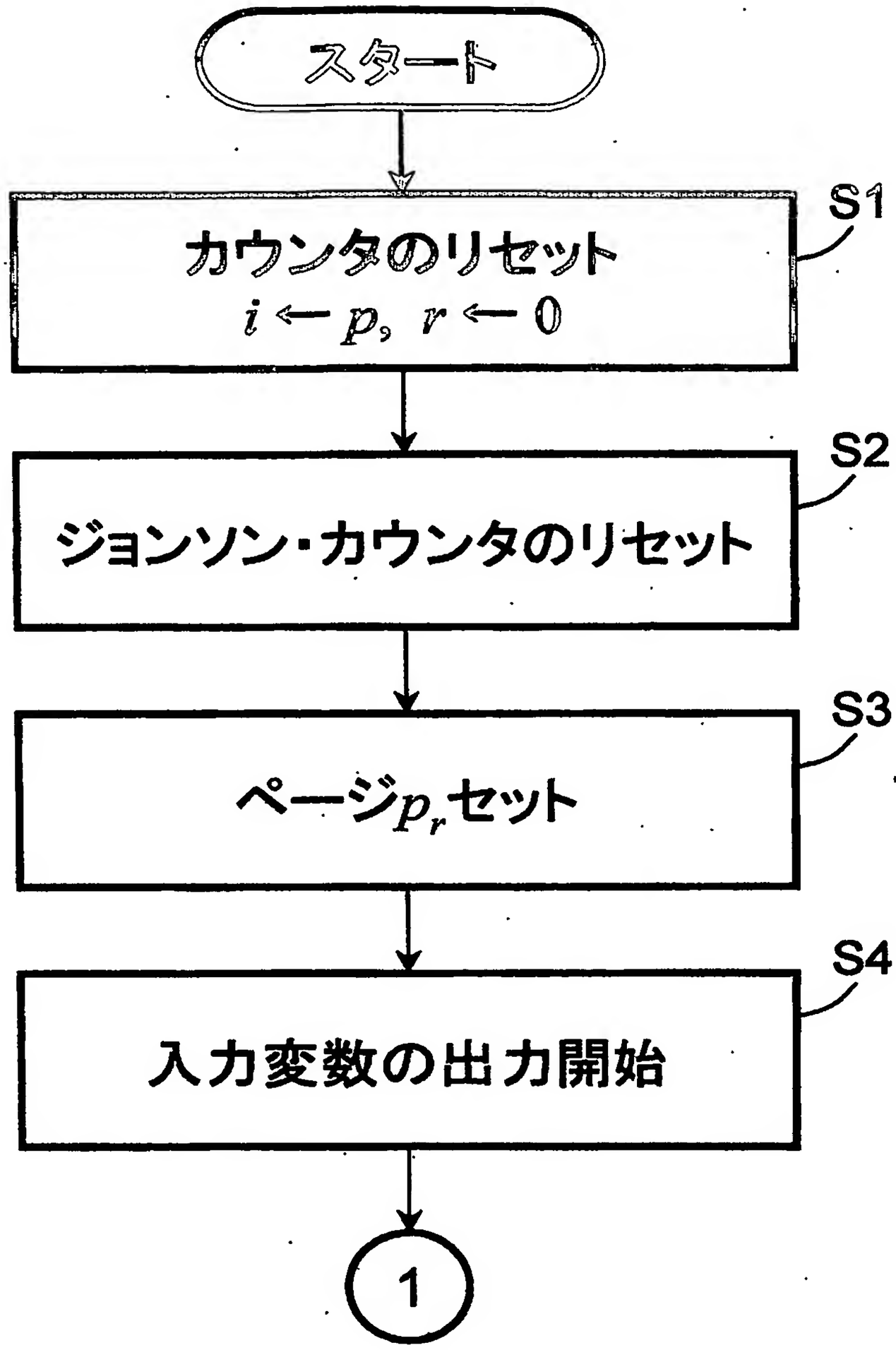
第２６図


6
接続メモリ
3
入力選択
メモリ
8
領域指定メモリ
4
$p(p_r)$
$CE_0 \sim CE_3$
論理関数
メモリ
$ADSP_0 \sim ADSP_3$
10
演算制御部
p_r
63
ページ・カウンタ
r
64
ADSP
出力コントローラ
END
62
ステップ・カウンタ
i
reset
clk
61
演算スタンバイコントロール
演算スタンバイ

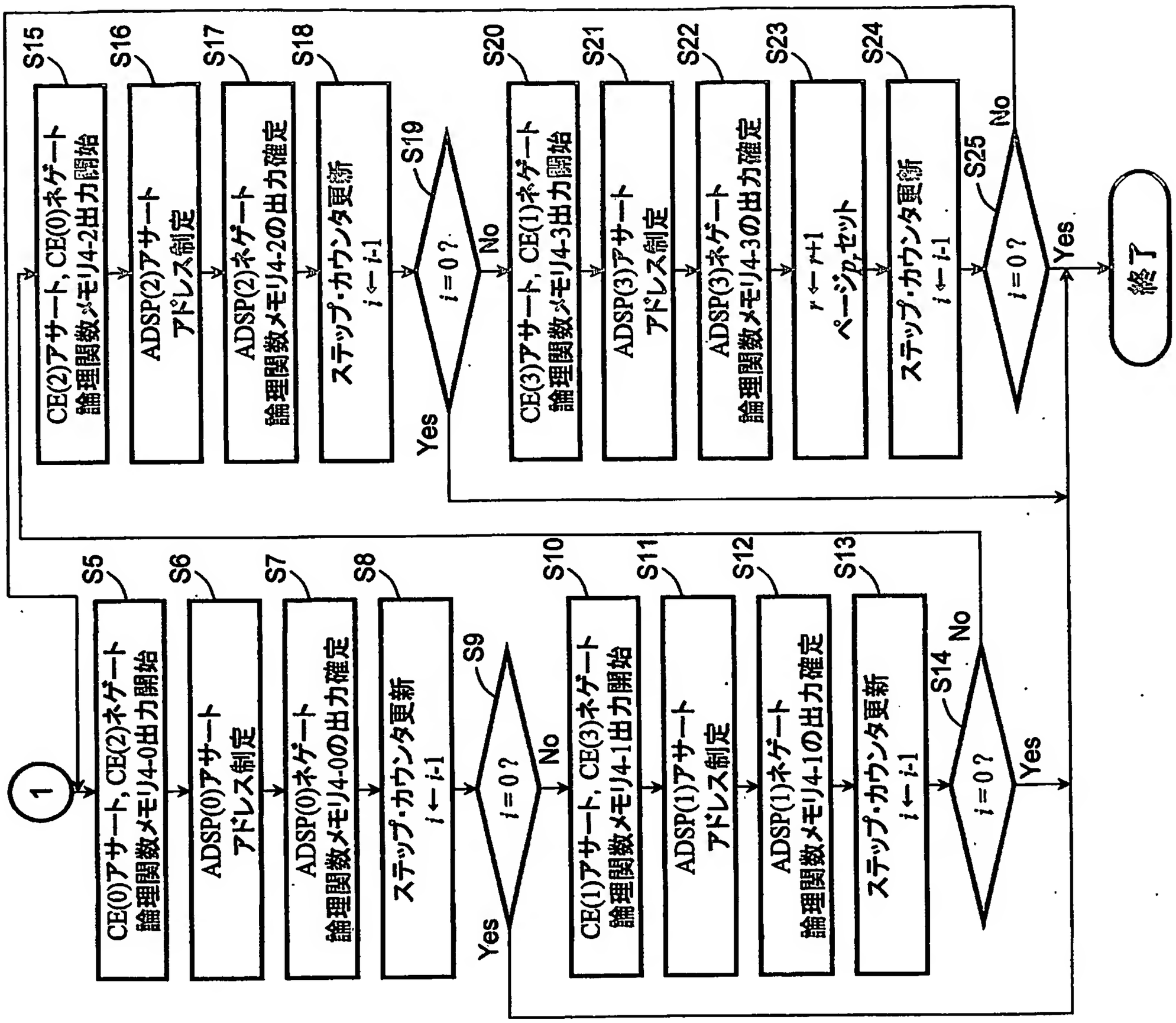
第２７図


64
69
68
67
66
65
70
71
72
73
74
CE_0
CE_1
CE_2
CE_3
$ADSP_0$
$ADSP_1$
$ADSP_2$
$ADSP_3$
Q_0
$\overline{Q}_0$
D_0
C
rst
reset'
clock
reset
END

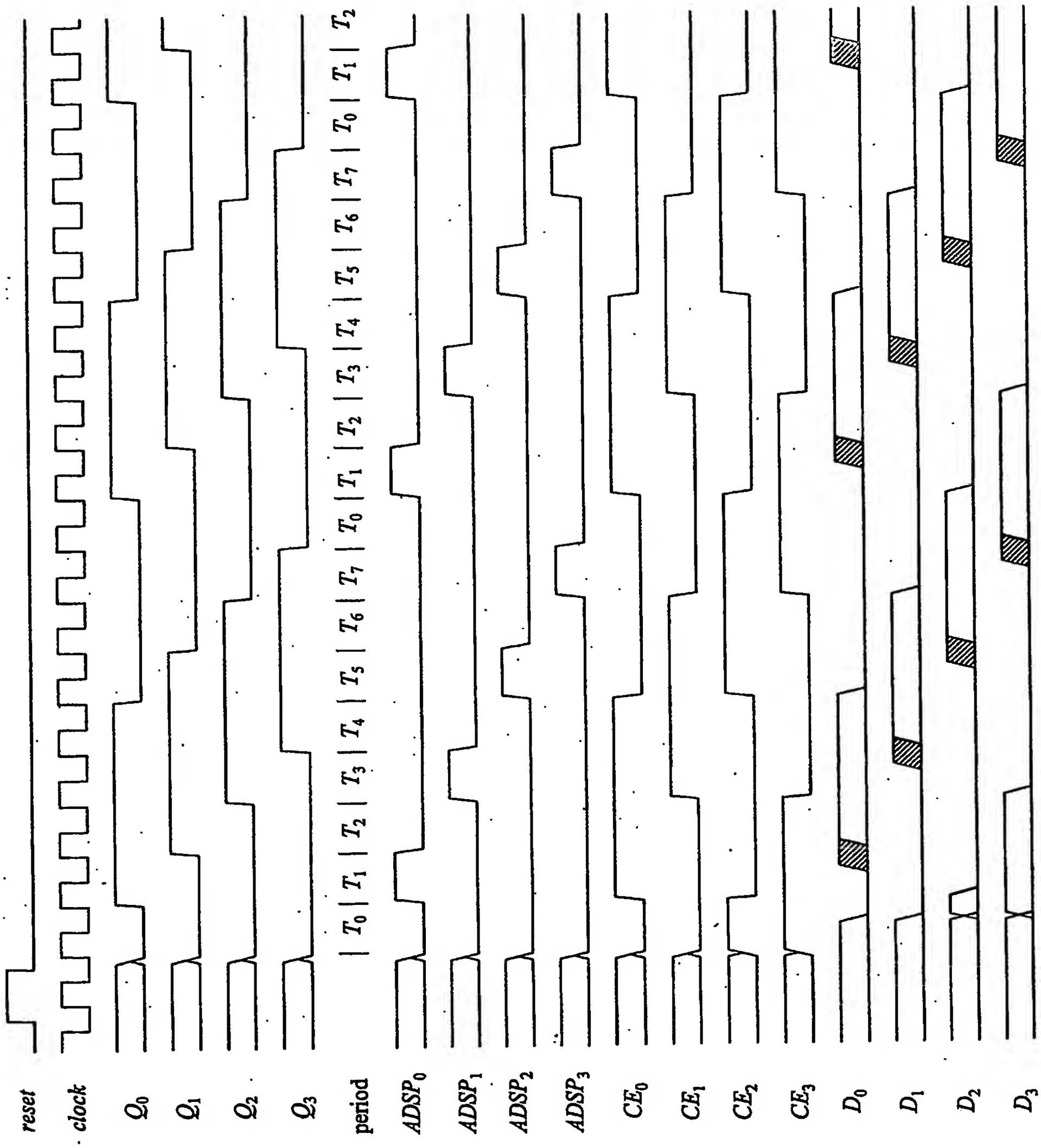
第２８（ａ）図


スタート
S1
カウンタのリセット
$i \leftarrow p,\ r \leftarrow 0$
S2
ジョンソン・カウンタのリセット
S3
ページp_rセット
S4
入力変数の出力開始
1

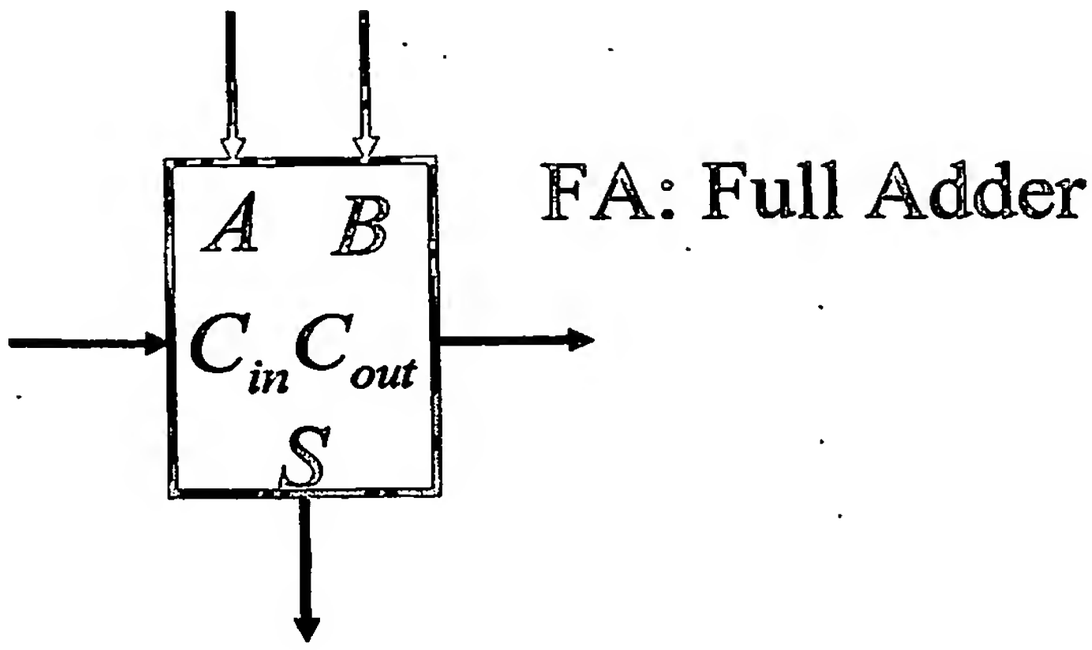
第２８（ｂ）図


S15
CE(2)アサート, CE(0)ネゲート
論理関数メモリ4-2出力開始
S16
ADSP(2)アサート
アドレス制定
S17
ADSP(2)ネゲート
論理関数メモリ4-2の出力確定
S18
ステップ・カウンタ更新
i ← i-1
S19
i = 0 ?
Yes
No
S20
CE(3)アサート, CE(1)ネゲート
論理関数メモリ4-3出力開始
S21
ADSP(3)アサート
アドレス制定
S22
ADSP(3)ネゲート
論理関数メモリ4-3の出力確定
S23
r ← r+1
ページD_rセット
S24
ステップ・カウンタ更新
i ← i-1
S25
i = 0 ?
No
Yes
終了
1
S5
CE(0)アサート, CE(2)ネゲート
論理関数メモリ4-0出力開始
S6
ADSP(0)アサート
アドレス制定
S7
ADSP(0)ネゲート
論理関数メモリ4-0の出力確定
S8
ステップ・カウンタ更新
i ← i-1
S9
i = 0 ?
Yes
No
S10
CE(1)アサート, CE(3)ネゲート
論理関数メモリ4-1出力開始
S11
ADSP(1)アサート
アドレス制定
S12
ADSP(1)ネゲート
論理関数メモリ4-1の出力確定
S13
ステップ・カウンタ更新
i ← i-1
S14
i = 0 ?
No
Yes

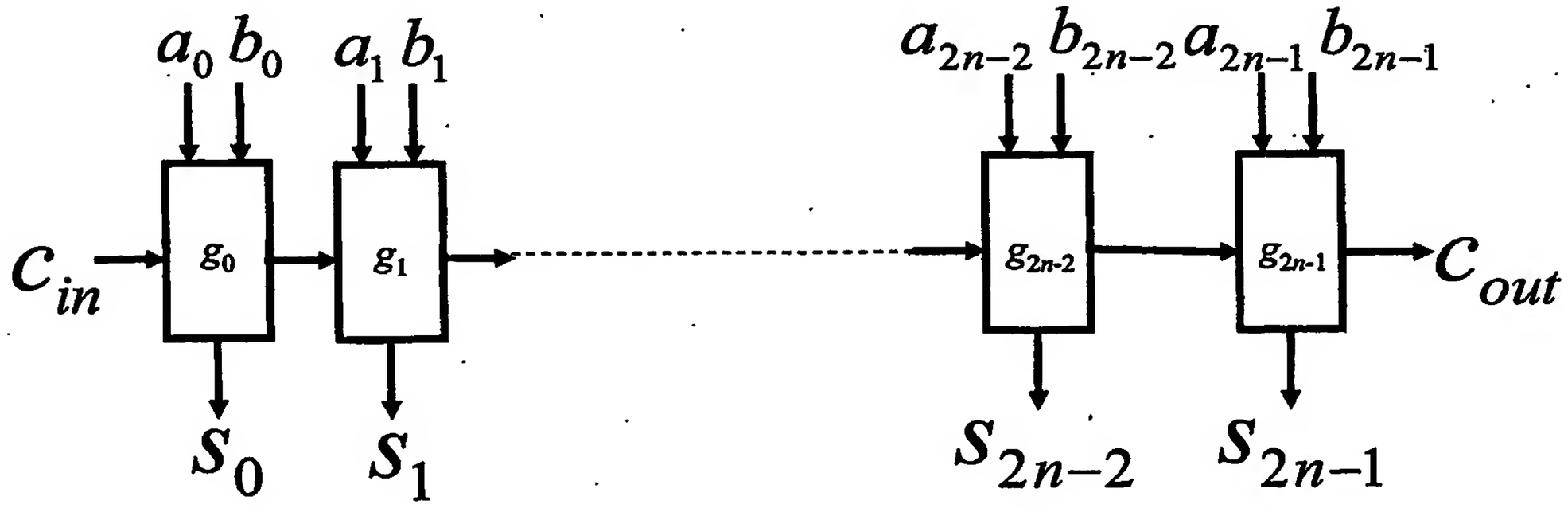
第２９図


reset
clock
Q_0
Q_1
Q_2
Q_3
period
$ADSP_0$
$ADSP_1$
$ADSP_2$
$ADSP_3$
CE_0
CE_1
CE_2
CE_3
D_0
D_1
D_2
D_3
T_0 | T_1 | T_2 | T_3 | T_4 | T_5 | T_6 | T_7 | T_0 | T_1 | T_2 | T_3 | T_4 | T_5 | T_6 | T_7 | T_0 | T_1 | T_2

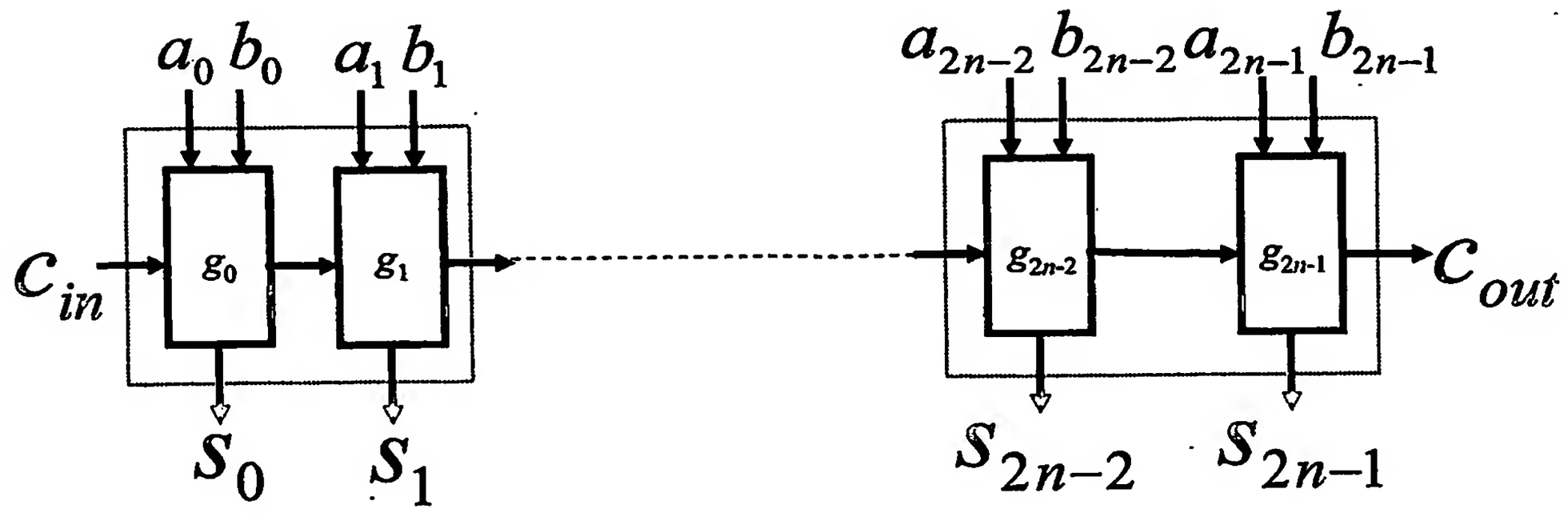
第３０図



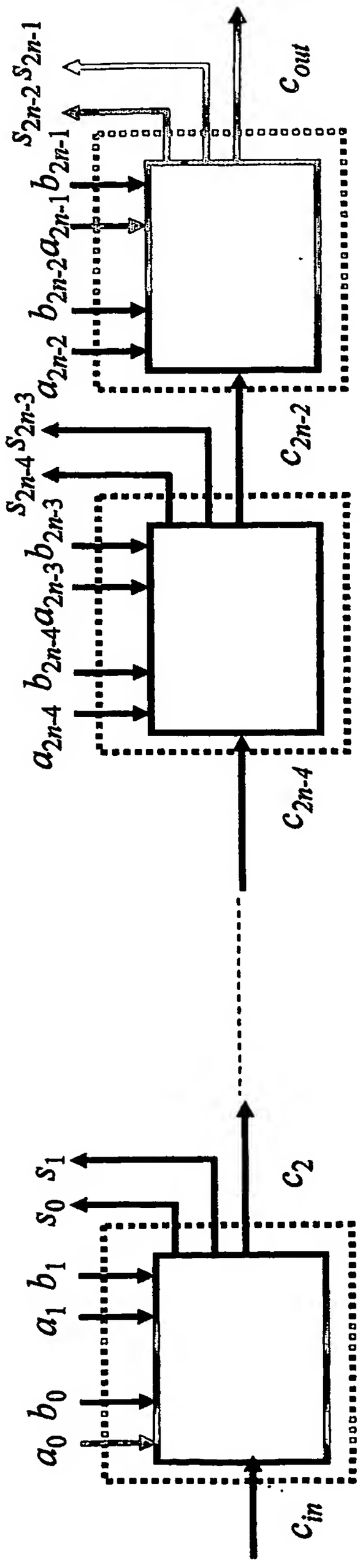
第３１図



第３２図



第３３図


s_{2n-2} s_{2n-1}
c_{out}
b_{2n-1}
a_{2n-1} b_{2n-2}
a_{2n-2} b_{2n-3}
a_{2n-3}
s_{2n-3} s_{2n-4}
c_{2n-2}
b_{2n-3}
a_{2n-3} b_{2n-4}
a_{2n-4}
c_{2n-4}
s_1 s_0
c_2
b_1
a_1 b_0
a_0
c_{in}

第３４図

c_{in}	a_1	a_0	b_1	b_0	c_{out}	s_1	s_0
1	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	0	0	1	1
1	0	1	0	1	0	1	0
1	0	1	1	0	0	1	0
1	0	1	1	1	1	0	1
1	1	0	0	0	0	1	1
1	1	0	0	1	0	1	0
1	1	0	1	0	0	1	0
1	1	0	1	1	1	0	1
1	1	1	0	0	0	1	1
1	1	1	0	1	1	0	0
1	1	1	1	0	1	0	0
1	1	1	1	1	1	0	1

c_{in}	a_1	a_0	b_1	b_0	c_{out}	s_1	s_0
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	1	0
0	0	0	1	1	0	1	1
0	0	1	0	0	0	1	0
0	0	1	0	1	0	1	1
0	0	1	1	0	0	1	1
0	0	1	1	1	1	0	0
0	1	0	0	0	0	1	0
0	1	0	0	1	0	1	1
0	1	0	1	0	0	1	1
0	1	0	1	1	1	0	0
0	1	1	0	0	0	1	1
0	1	1	0	1	1	0	0
0	1	1	1	0	1	0	0
0	1	1	1	1	1	0	1

第３５図

		i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
出力値	o_0	1	1	1	1	1	1	1	1
	o_1	1	1	1	1	1	1	1	1
	o_2	1	0	1	0	0	1	0	1
	o_3	0	1	1	0	1	1	0	0
	o_4	0	0	0	1	0	0	1	1
	o_5	1	1	1	1	1	1	1	1
	o_6	1	1	1	1	1	1	1	1
	o_7	1	1	1	1	1	1	1	1
アドレス	i_0	1	1	1	1	1	1	1	1
	i_1	0	0	0	0	0	0	0	0
	i_2	0	0	0	0	1	1	1	1
	i_3	0	0	1	1	0	0	1	1
	i_4	0	1	0	1	0	1	0	1
	i_5	1	1	1	1	1	1	1	1
	i_6	1	1	1	1	1	1	1	1
	i_7	1	1	1	1	1	1	1	1

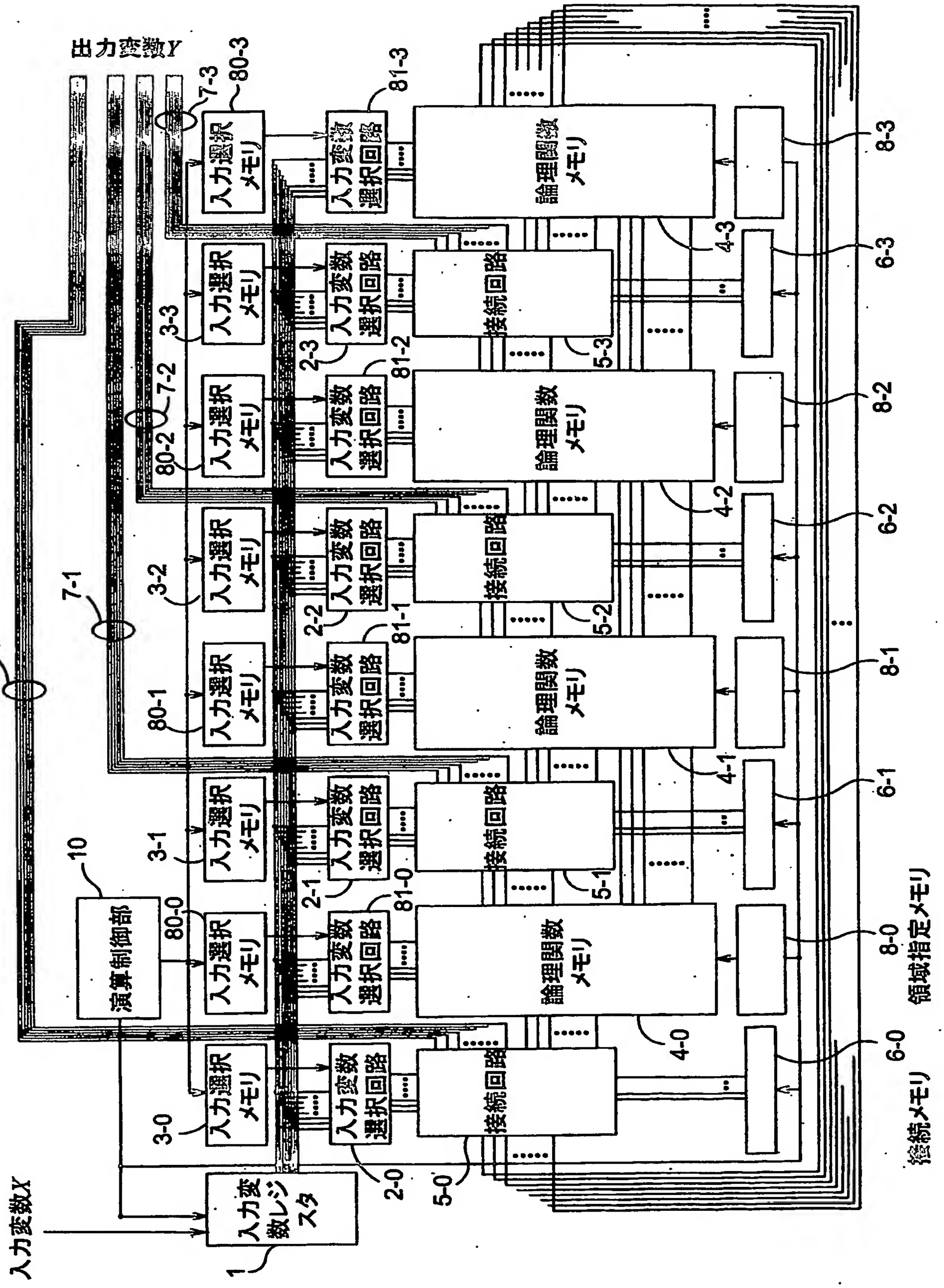
		i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
出力値	o_0	1	1	1	1	1	1	1	1
	o_1	1	1	1	1	1	1	1	1
	o_2	0	1	0	1	1	0	1	0
	o_3	0	0	1	1	0	1	1	1
	o_4	0	0	0	1	0	0	1	1
	o_5	1	1	1	1	1	1	1	1
	o_6	1	1	1	1	1	1	1	1
	o_7	1	1	1	1	1	1	1	1
アドレス	i_0	0	0	0	0	0	0	0	0
	i_1	0	0	0	0	0	0	0	0
	i_2	0	0	0	0	1	1	1	1
	i_3	0	0	1	1	0	0	1	1
	i_4	0	1	0	1	0	1	0	1
	i_5	1	1	1	1	1	1	1	1
	i_6	1	1	1	1	1	1	1	1
	i_7	1	1	1	1	1	1	1	1

第３６図

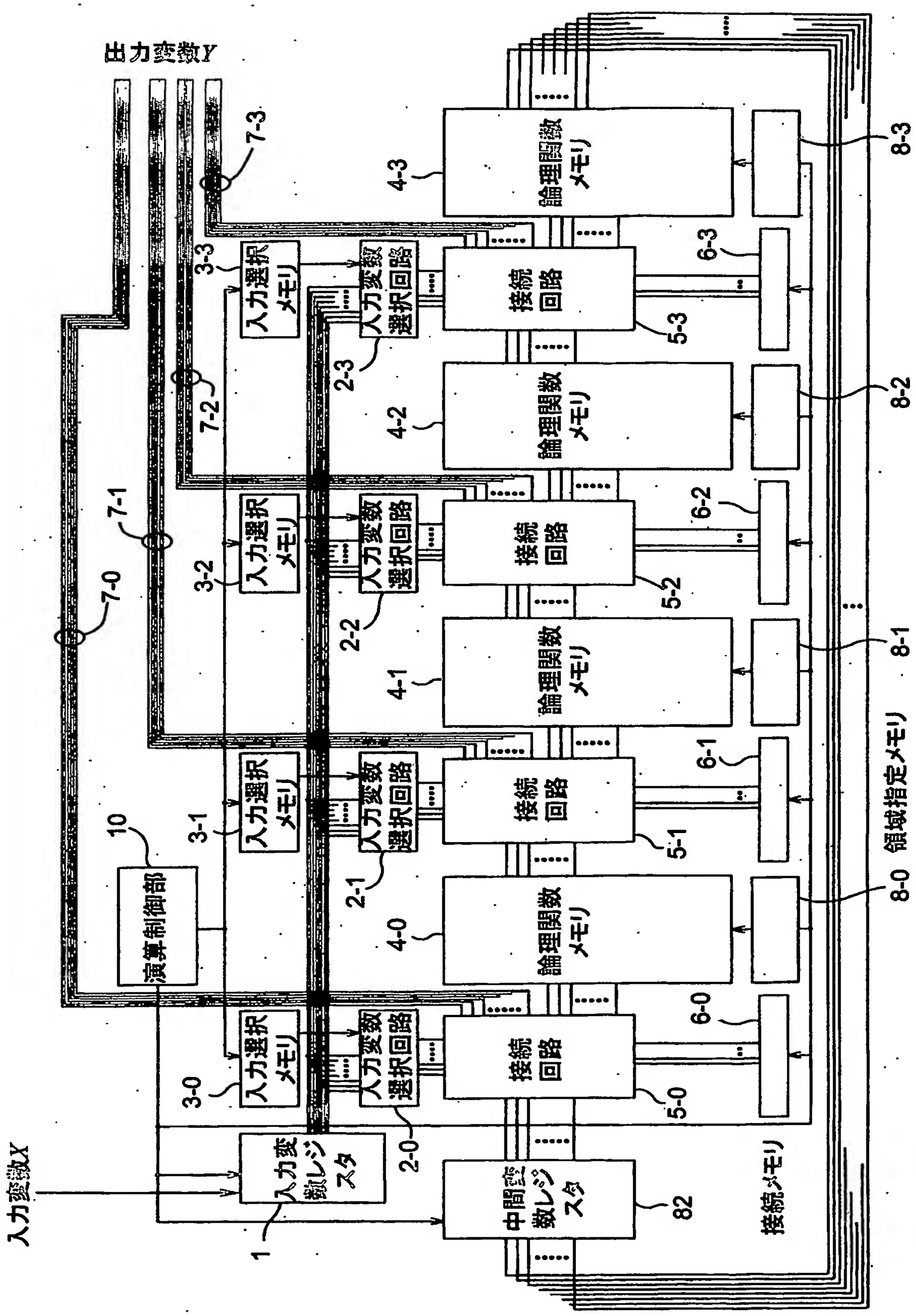
出力値																
	o_0	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_1	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_2	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1
	o_3	0	1	1	0	1	1	0	0	1	0	0	1	0	0	1	1
	o_4	0	0	0	1	0	0	1	1	0	1	1	1	1	1	1	1
	o_5	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_6	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_7	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
アドレス	i_0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	i_1	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	i_2	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	i_3	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	i_4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	i_5	0	0	0	0	0	1	1	1	0	1	0	1	1	1	1	1
	i_6	0	0	0	1	1	0	0	1	1	0	0	1	0	1	1	1
	i_7	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	1

出力値																
	o_0	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_1	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_2	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0
	o_3	0	0	1	1	0	1	1	0	1	1	0	0	1	0	1	1
	o_4	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1
	o_5	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_6	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	o_7	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
アドレス	i_0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	i_1	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	i_2	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	i_3	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
	i_4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	i_5	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	i_6	0	0	1	1	0	0	1	1	0	0	1	1	0	1	1	1
	i_7	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

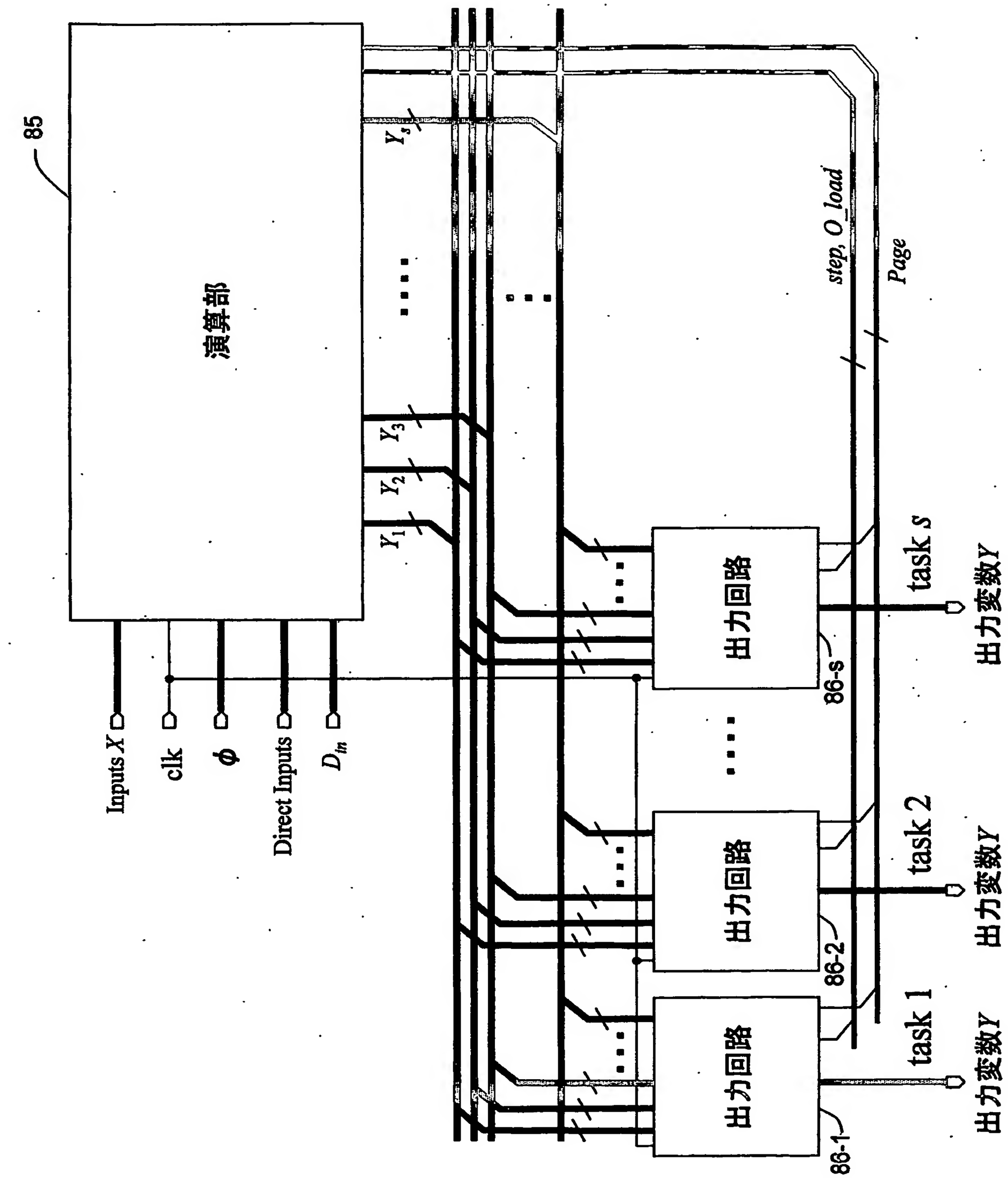
第３７図


出力変数Y
入力変数X
演算制御部
10
入力変数レジスタ
1
入力選択メモリ
入力変数選択回路
接続回路
論理関数メモリ
7-0
7-1
7-2
7-3
80-0
80-1
80-2
80-3
81-0
81-1
81-2
81-3
2-0
2-1
2-2
2-3
3-0
3-1
3-2
3-3
4-0
4-1
4-2
4-3
5-0
5-1
5-2
5-3
6-0
6-1
6-2
6-3
8-0
8-1
8-2
8-3
接続メモリ
領域指定メモリ

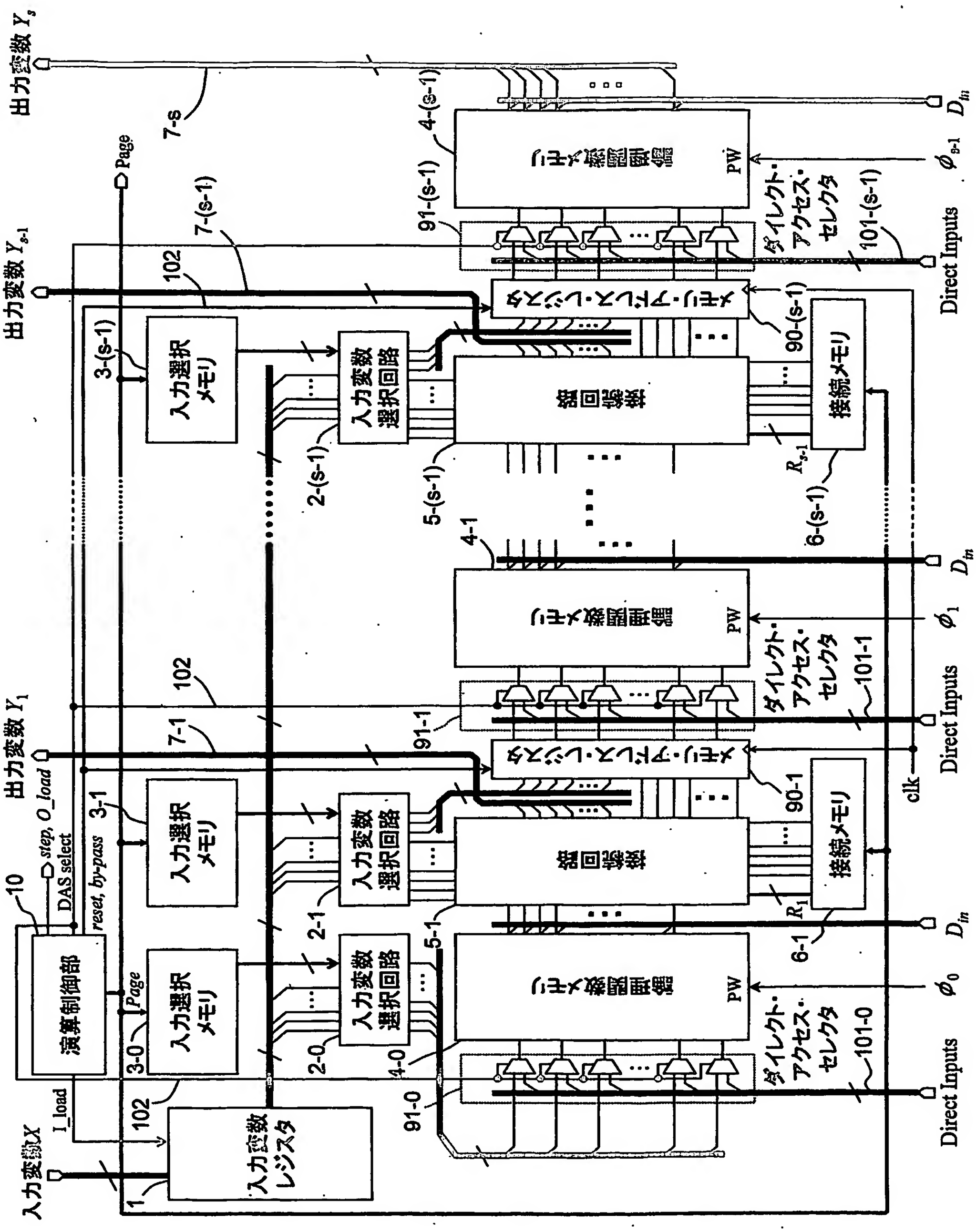
第３８図


出力変数Y
7-3
4-3
論理関数メモリ
8-3
3-3
入力選択メモリ
入力変数選択回路
接続回路
6-3
5-3
2-3
7-2
4-2
論理関数メモリ
8-2
7-1
7-0
3-2
入力選択メモリ
入力変数選択回路
接続回路
6-2
5-2
2-2
4-1
論理関数メモリ
8-1
10
3-1
入力選択メモリ
入力変数選択回路
接続回路
6-1
5-1
2-1
4-0
論理関数メモリ
8-0
演算制御部
3-0
入力選択メモリ
入力変数選択回路
接続回路
6-0
5-0
2-0
入力変数X
入力変数レジスタ
1
中間変数レジスタ
82
接続メモリ
領域指定メモリ

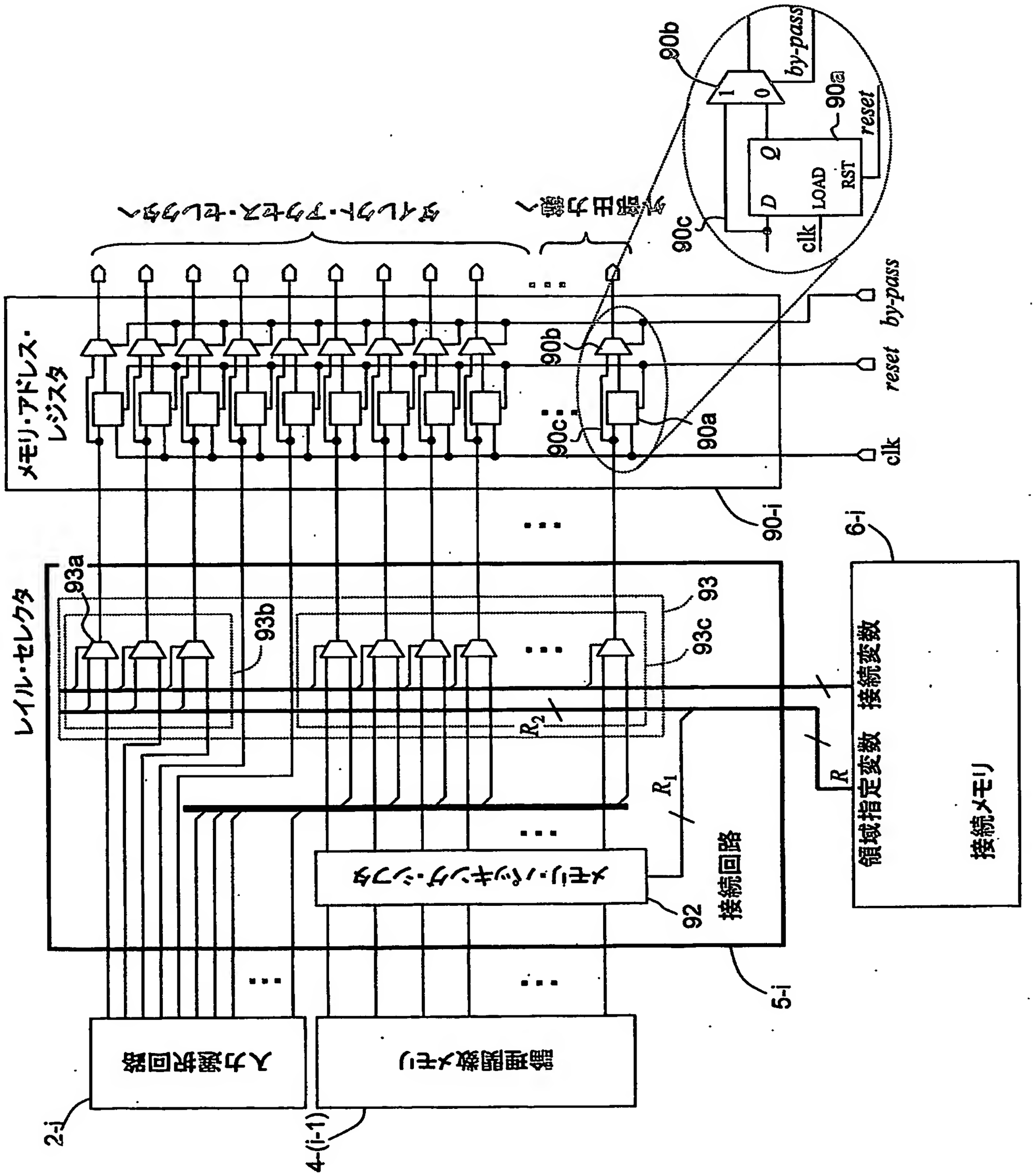
第３９図


85
演算部
Y_s
Y_3
Y_2
Y_1
step, O_load
Page
task s
出力変数Y
出力回路
86-s
Input Inputs X
clk
ϕ
Direct Inputs
D_{in}
出力回路
86-2
task 2
出力変数Y
出力回路
86-1
task 1
出力変数Y

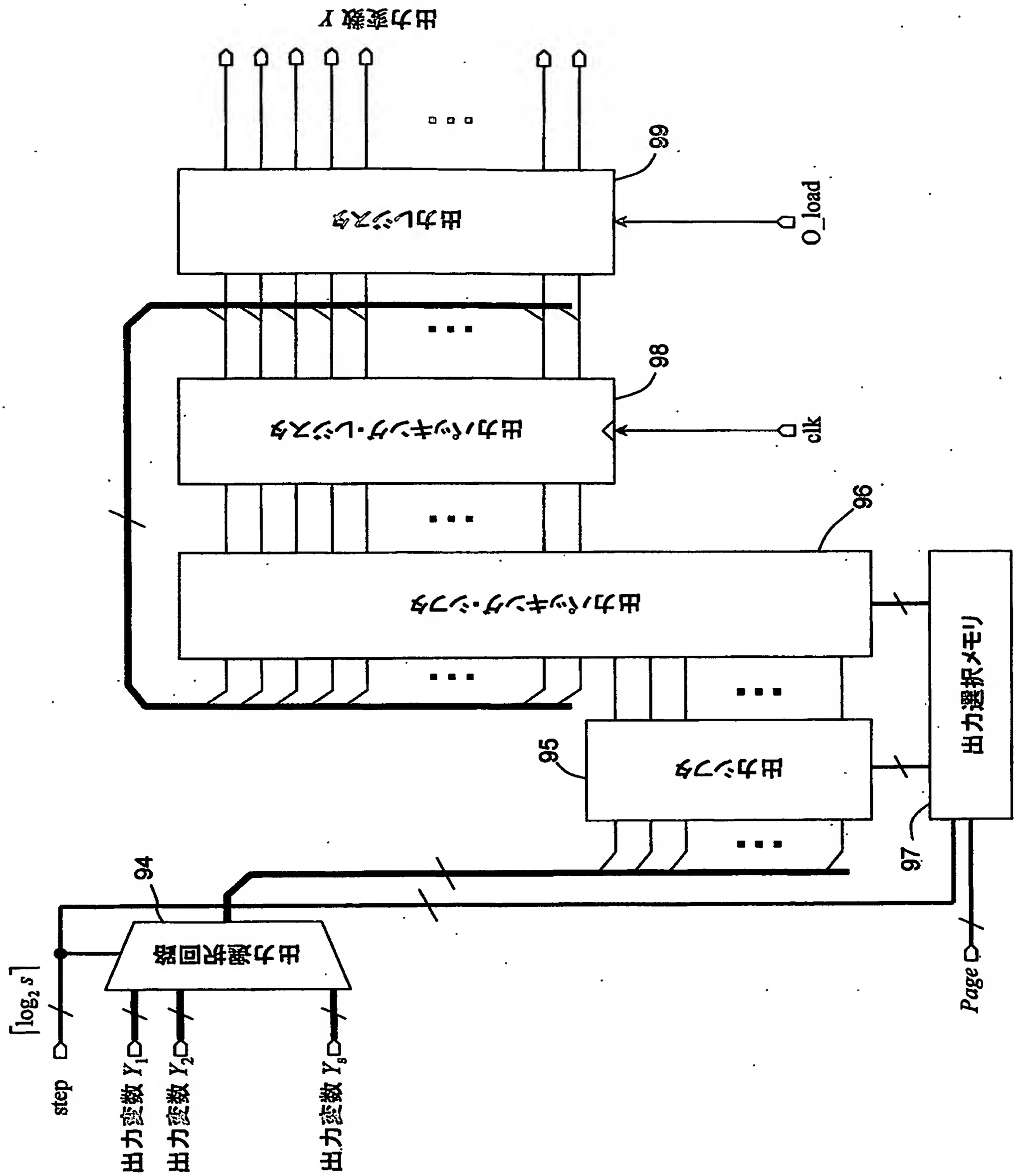
第４０図


出力変数 Y_s
出力変数 Y_{s-1}
出力変数 Y_1
出力変数 Y_0
入力変数 X
Page
step, O_load
DAS select
reset, by-pass
I_load
clk
7-s
7-(s-1)
7-1
102
10
3-(s-1)
3-1
3-0
入力選択メモリ
2-(s-1)
2-1
2-0
入力変数選択回路
5-(s-1)
5-1
5-0
接続回路
90-(s-1)
90-1
90-0
接続メモリ
R_{s-1}
R_1
R_0
6-(s-1)
6-1
6-0
4-(s-1)
4-1
4-0
論理関数メモリ
PW
91-(s-1)
91-1
91-0
メモリ・アドレス・レジスタ
ダイレクト・アクセス・セレクタ
101-(s-1)
101-1
101-0
Direct Inputs
D_{in}
ϕ_{s-1}
ϕ_1
ϕ_0
演算制御部
入力変数レジスタ
1

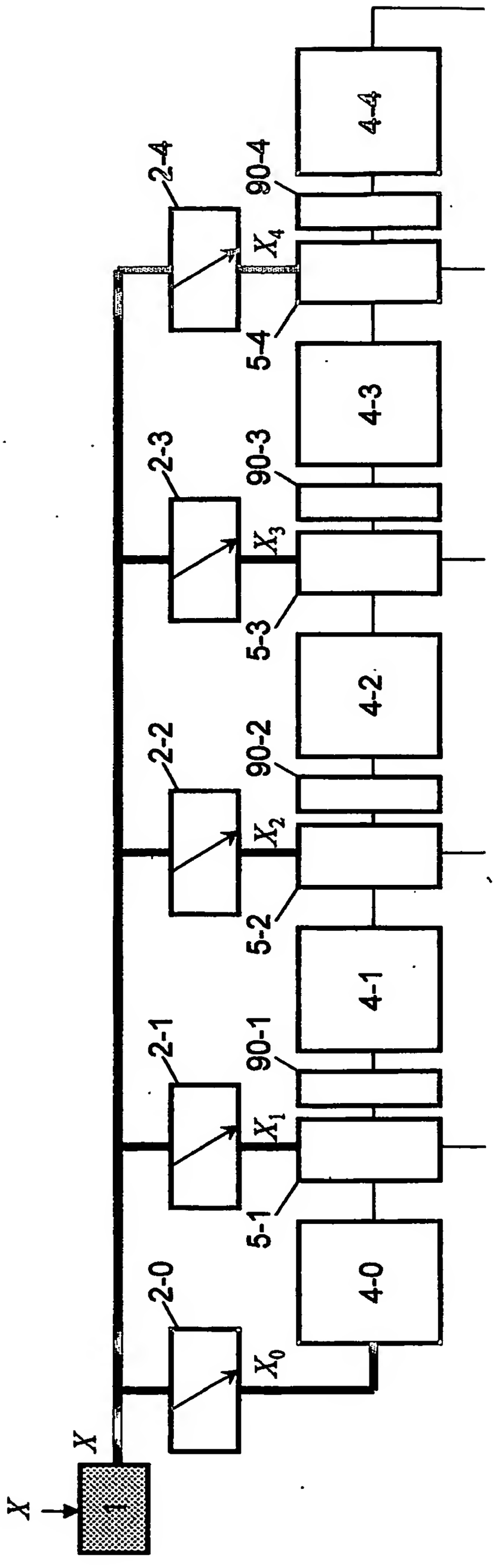
第４１図


90b
by-pass
1
0
90a
reset
Q
D
LOAD
RST
clk
90c
メモリ・アドレス・レジスタ
90b
90c
90a
clk
reset
by-pass
90-i
6-i
93a
93b
93
93c
レイル・セレクタ
R_2
R_1
R
領域指定変数
接続変数
接続メモリ
メモリ・インタフェース
92
接続回路
5-i
入力選択回路
論理関数メモリ
2-i
4-(i-1)

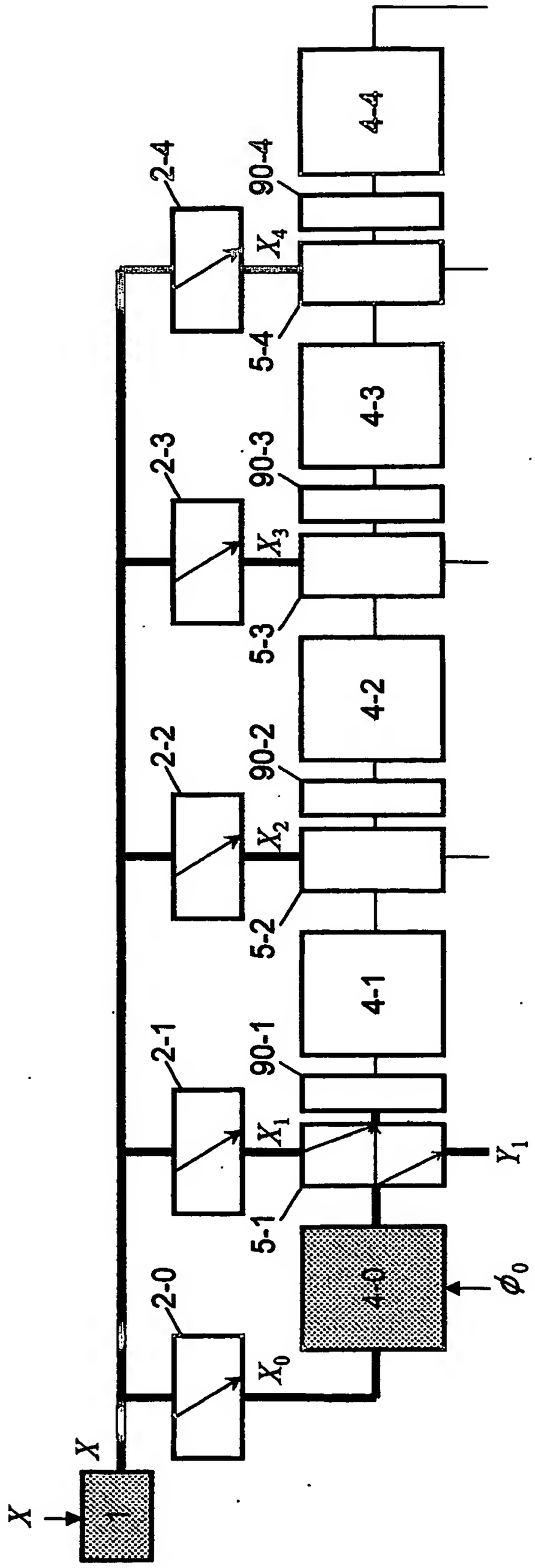
第４２図


出力変数 Y
99
出力レジスタ
O_load
98
出力ラッチ・レジスタ
clk
96
出力ラッチ・ラッチ
95
出力ラッチ
出力選択メモリ
97
94
出力選択回路
$\lceil \log_2 s \rceil$
step
出力変数 Y_1
出力変数 Y_2
出力変数 Y_s
Page

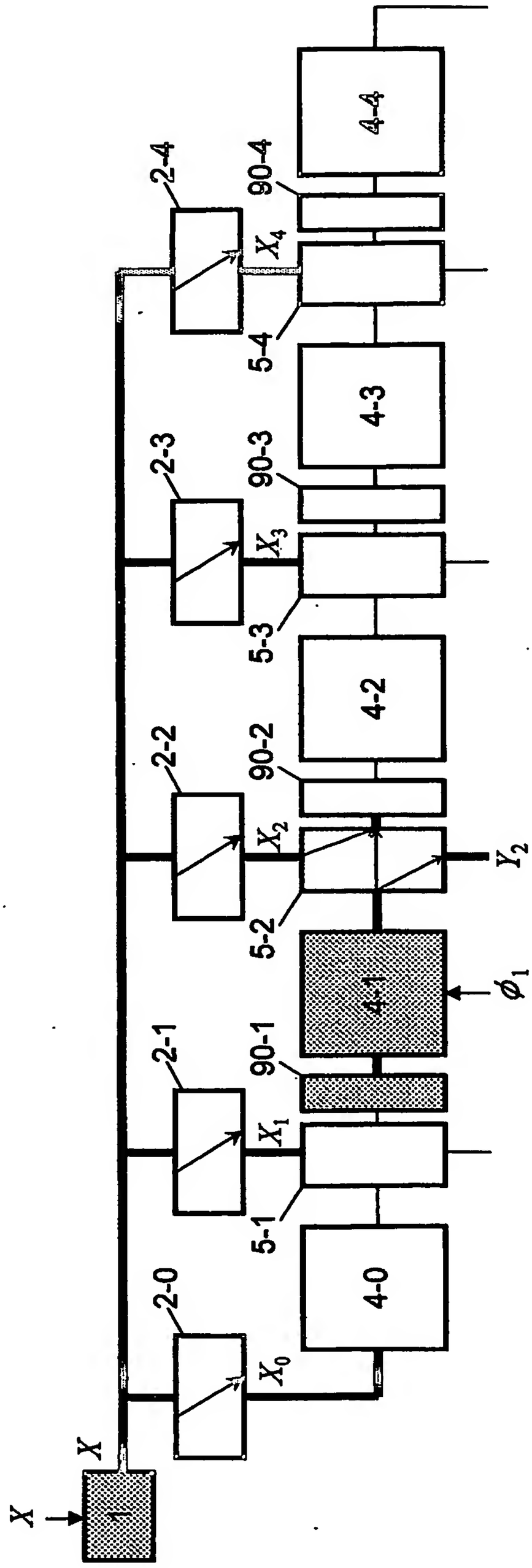
第４３（ａ）図


X
1
X
2-0
X0
5-1
90-1
X1
2-1
4-0
5-2
90-2
X2
2-2
4-1
5-3
90-3
X3
2-3
4-2
5-4
90-4
X4
2-4
4-3
4-4

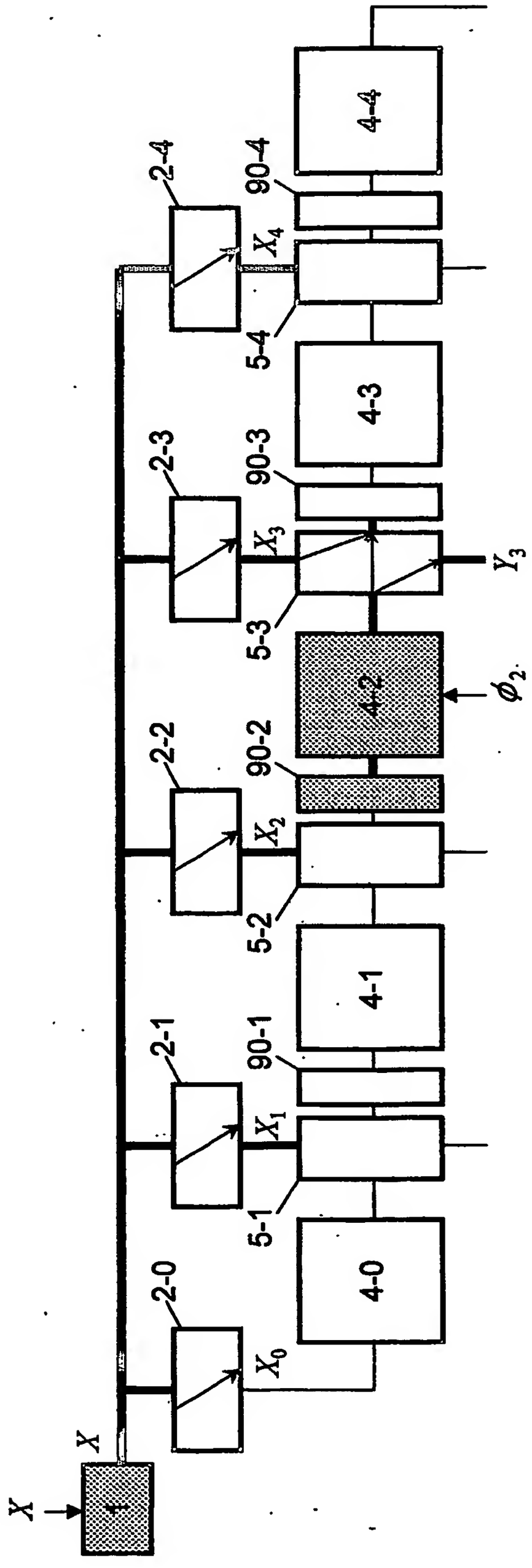
第４３（ｂ）図


X
1
2-0
2-1
2-2
2-3
2-4
4-0
4-1
4-2
4-3
4-4
5-1
5-2
5-3
5-4
90-1
90-2
90-3
90-4
X_0
X_1
X_2
X_3
X_4
Y_1
ϕ_0

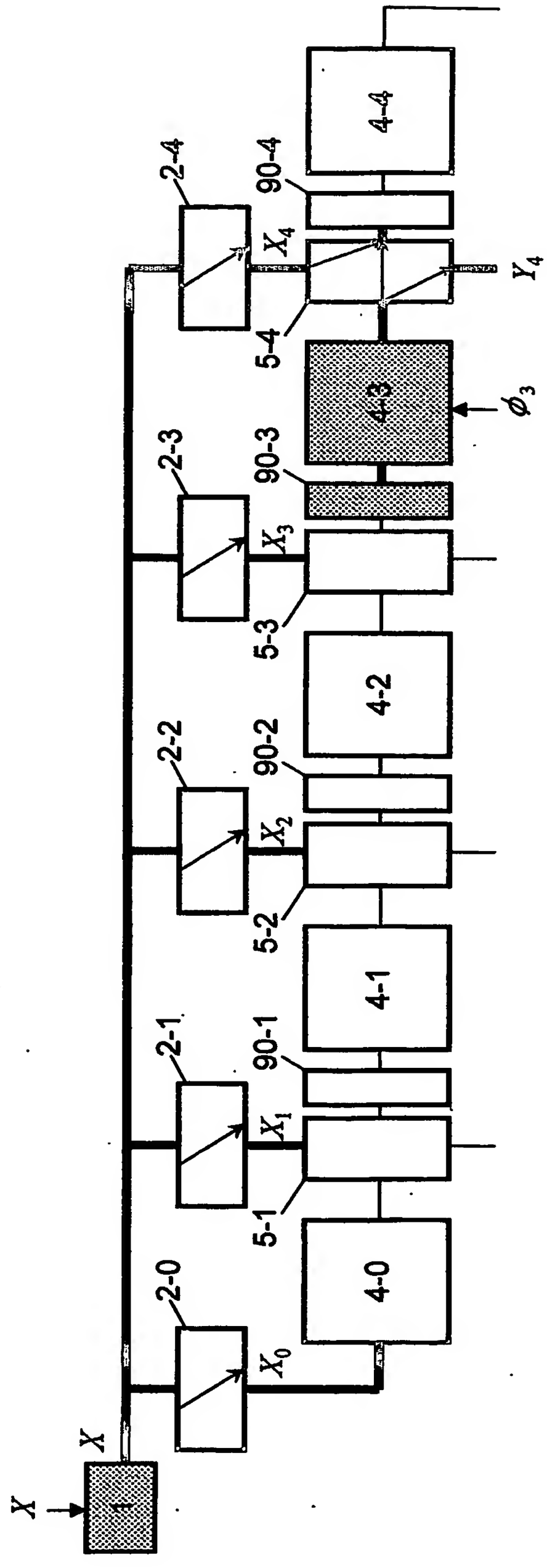
第４４（ａ）図


4-4
2-4
90-4
X_4
5-4
4-3
2-3
90-3
X_3
5-3
4-2
2-2
90-2
X_2
5-2
Y_2
4-1
ϕ_1
2-1
90-1
X_1
5-1
4-0
2-0
X_0
X
1
X

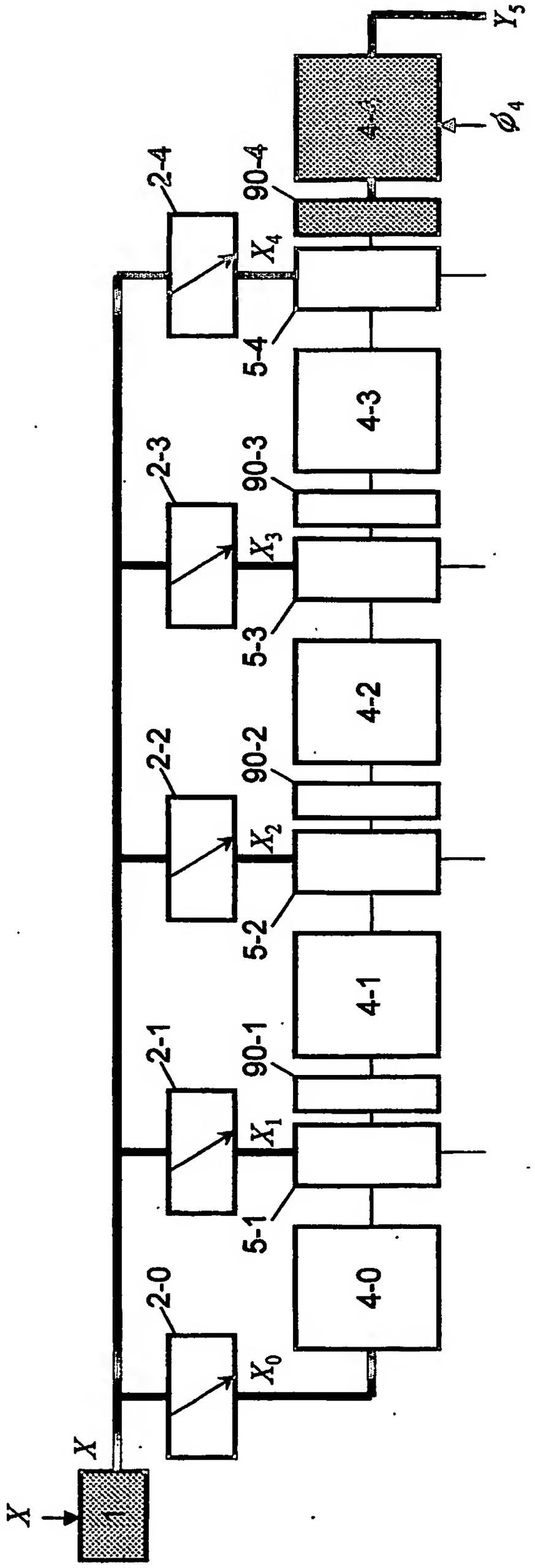
第４４（ｂ）図


4-4
90-4
2-4
X_4
5-4
4-3
90-3
2-3
X_3
Y_3
5-3
4-2
ϕ_2
90-2
2-2
X_2
5-2
4-1
90-1
2-1
X_1
5-1
4-0
2-0
X_0
X
X
1

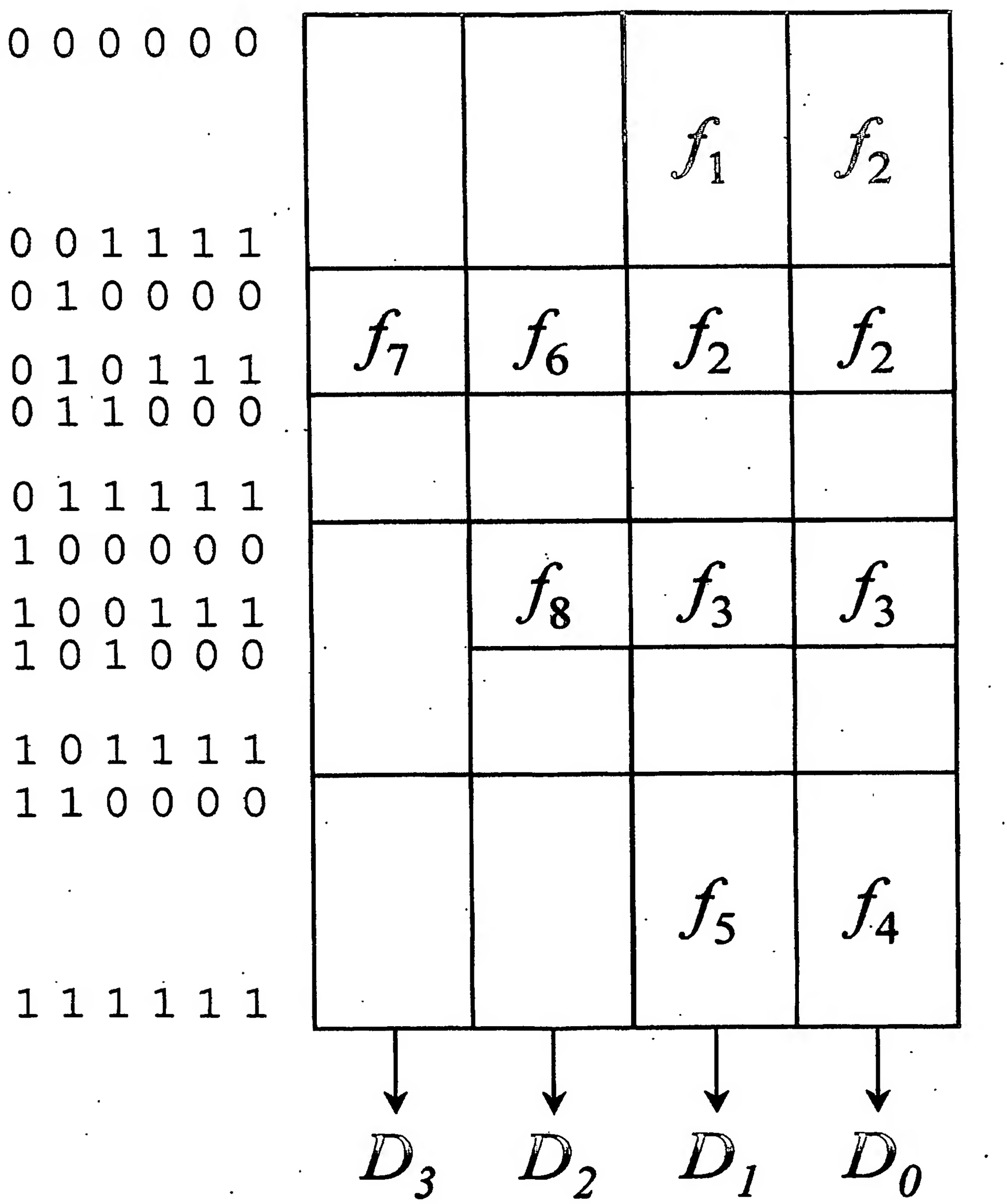
第４５（ａ）図


4-4
2-4
90-4
X_4
Y_4
5-4
4-3
ϕ_3
2-3
90-3
X_3
5-3
4-2
2-2
90-2
X_2
5-2
4-1
2-1
90-1
X_1
5-1
4-0
2-0
X_0
X
X
1

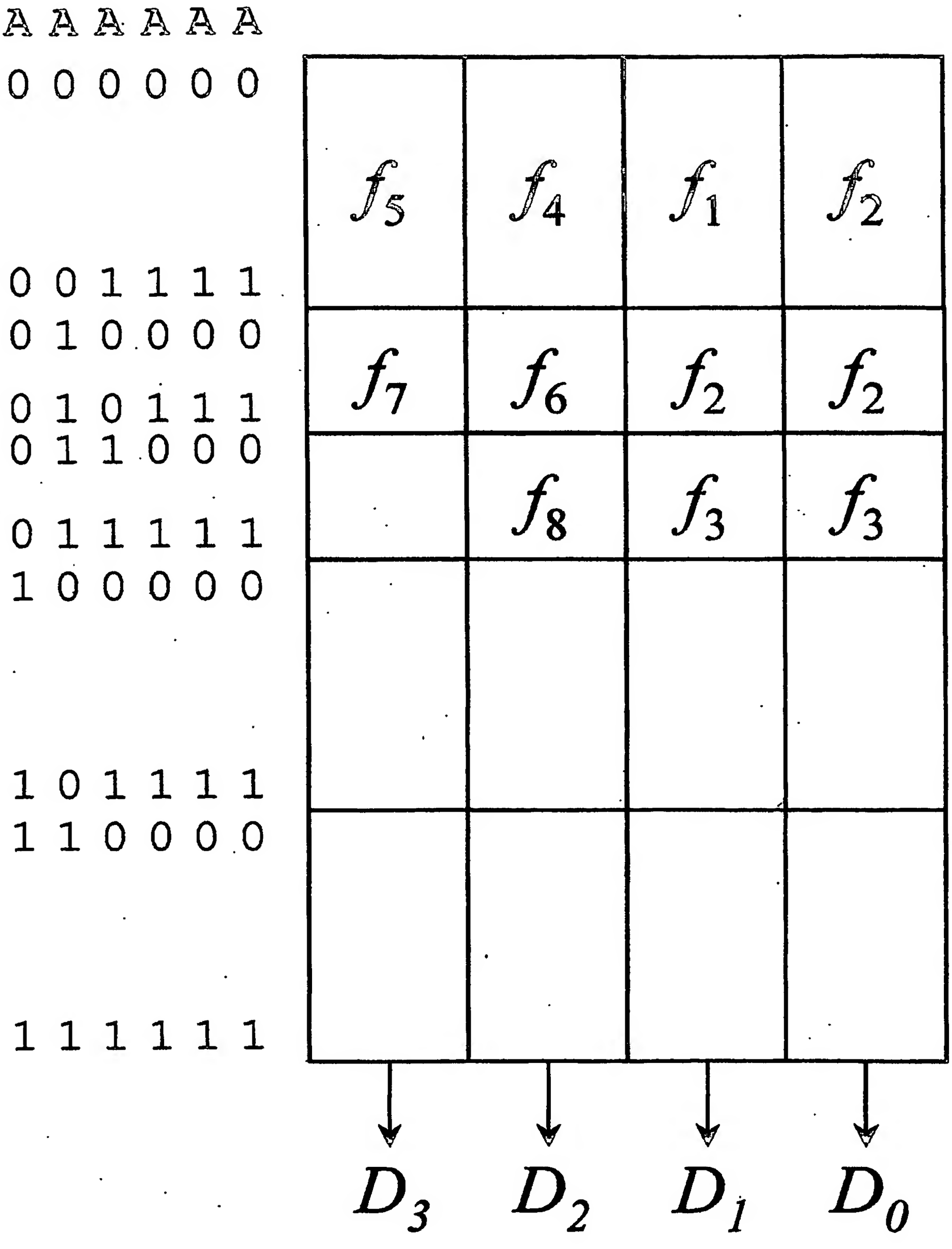
第４５（ｂ）図


X
1
2-0
2-1
2-2
2-3
2-4
X0
X1
X2
X3
X4
4-0
4-1
4-2
4-3
5-1
5-2
5-3
5-4
90-1
90-2
90-3
90-4
Y5
φ4

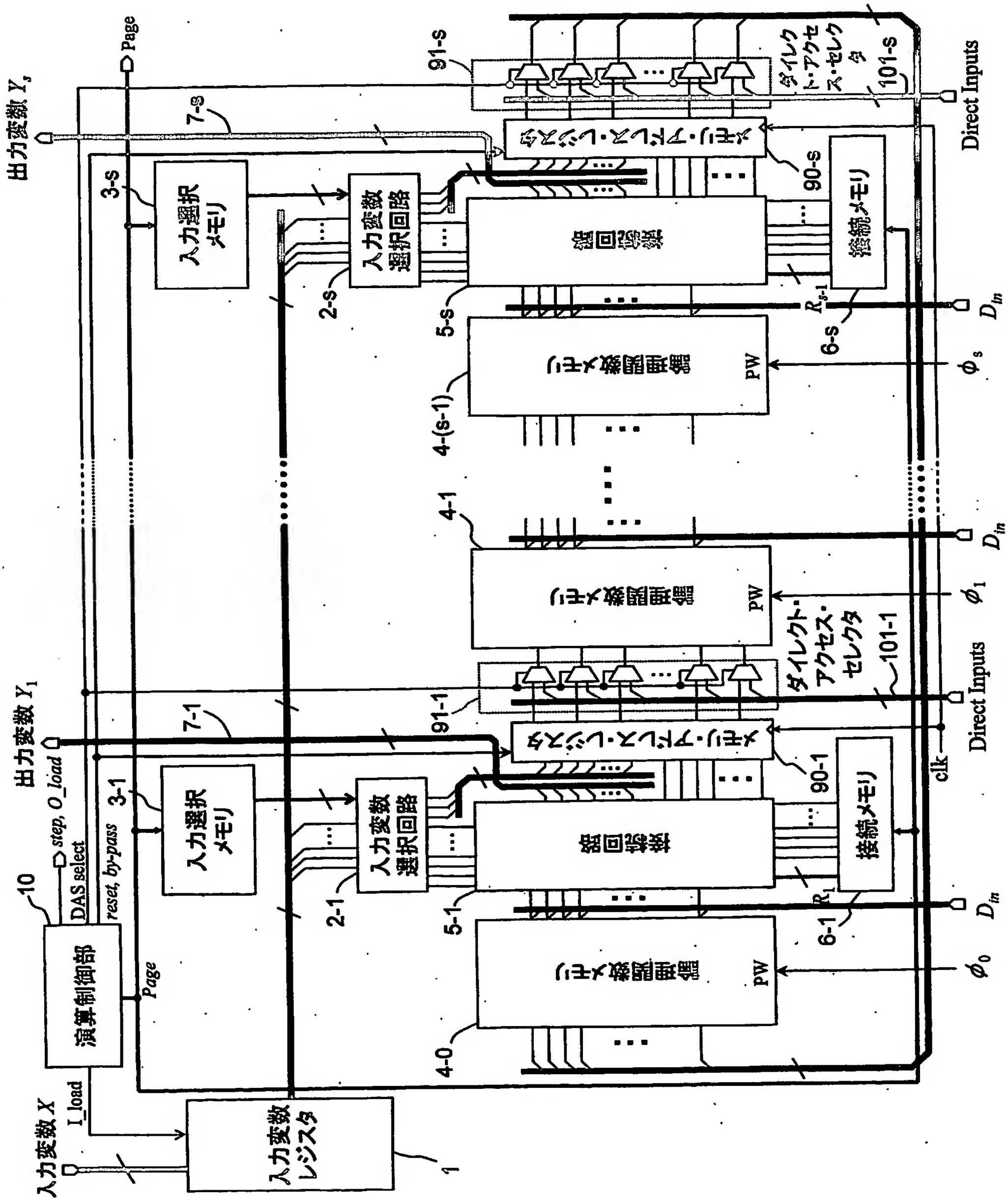
第４６（ａ）図


A A A A A A
0 0 0 0 0 0
0 0 1 1 1 1
0 1 0 0 0 0
0 1 0 1 1 1
0 1 1 0 0 0
0 1 1 1 1 1
1 0 0 0 0 0
1 0 0 1 1 1
1 0 1 0 0 0
1 0 1 1 1 1
1 1 0 0 0 0
1 1 1 1 1 1
f_1
f_2
f_7
f_6
f_2
f_2
f_8
f_3
f_3
f_5
f_4
D_3
D_2
D_1
D_0

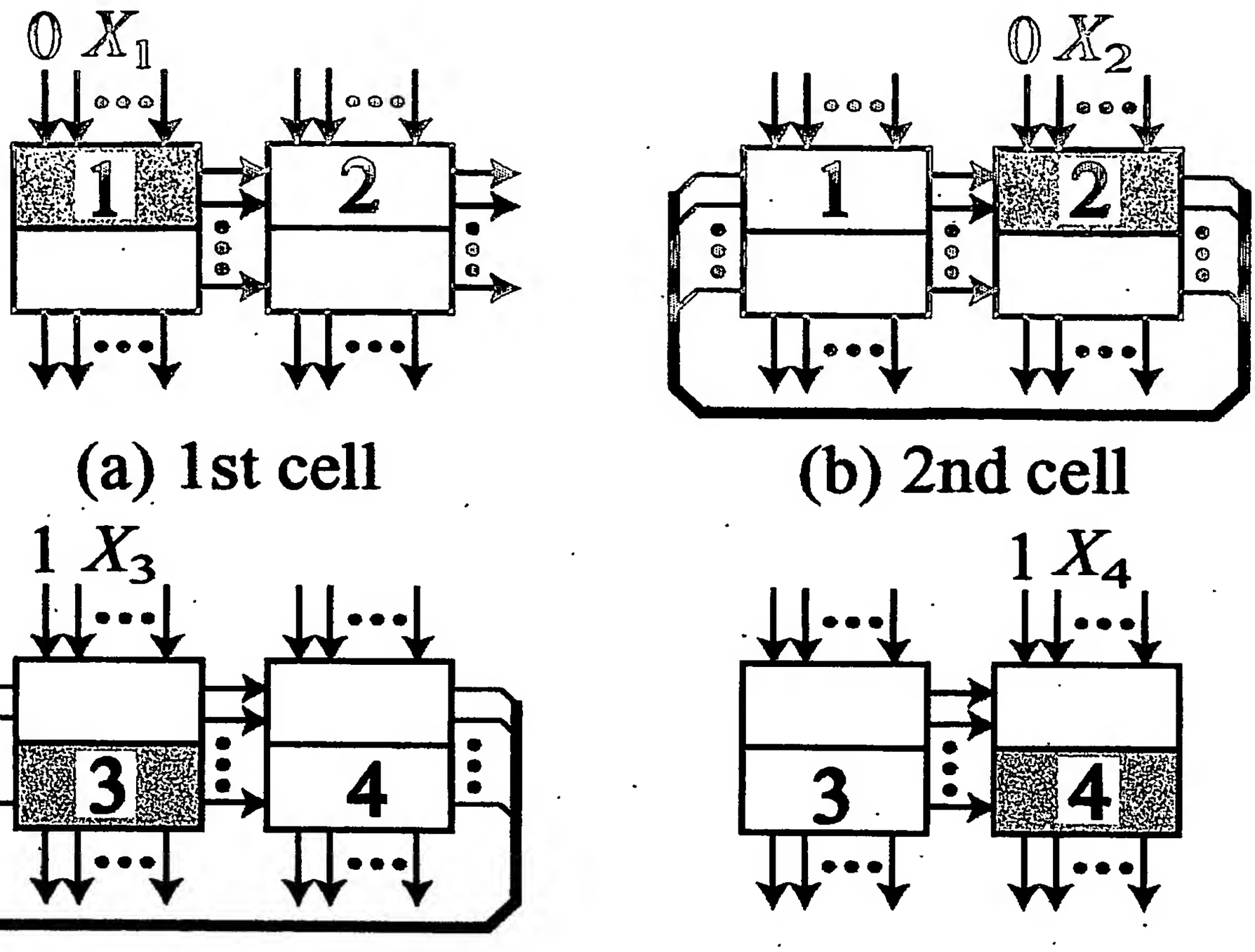
第４６（ｂ）図


A A A A A A
0 0 0 0 0 0
0 0 1 1 1 1
0 1 0 0 0 0
0 1 0 1 1 1
0 1 1 0 0 0
0 1 1 1 1 1
1 0 0 0 0 0
1 0 1 1 1 1
1 1 0 0 0 0
1 1 1 1 1 1
f_5 f_4 f_1 f_2
f_7 f_6 f_2 f_2
f_8 f_3 f_3
D_3 D_2 D_1 D_0

第４７図


出力変数 Y_s
Page
91-s
7-s
ダイレクト・アクセス・セレクタ
101-s
Direct Inputs
3-s
入力選択メモリ
入力変数選択回路
90-s
接続メモリ
2-s
5-s
R_{s-1}
6-s
D_{in}
ϕ_s
4-(s-1)
PW
4-1
ϕ_1
出力変数 Y_1
91-1
7-1
90-1
clk
3-1
2-1
5-1
R_1
6-1
ϕ_0
4-0
10
step, O_load
DAS select
reset, by-pass
演算制御部
入力変数 X
I_load
入力変数レジスタ
1

第４８図



(a) 1st cell

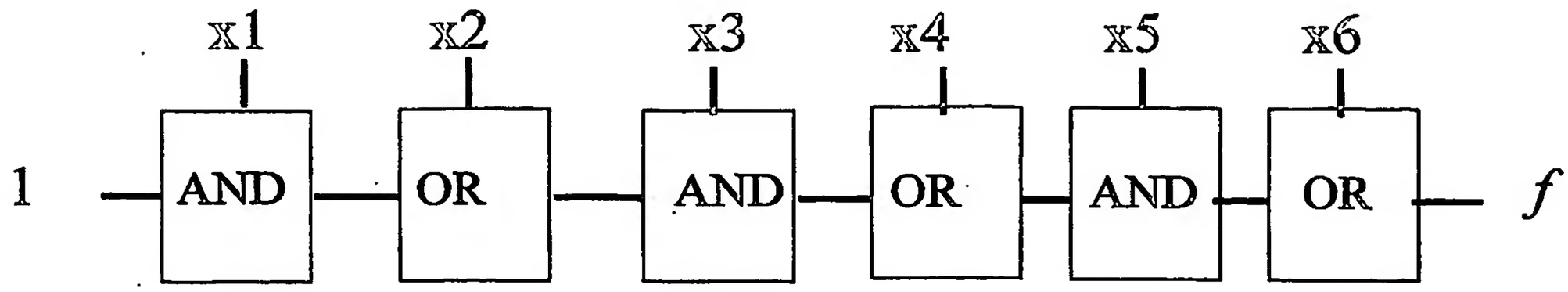
(b) 2nd cell

(c) 3rd cell

(d) 4th cell

第４９（ａ）図

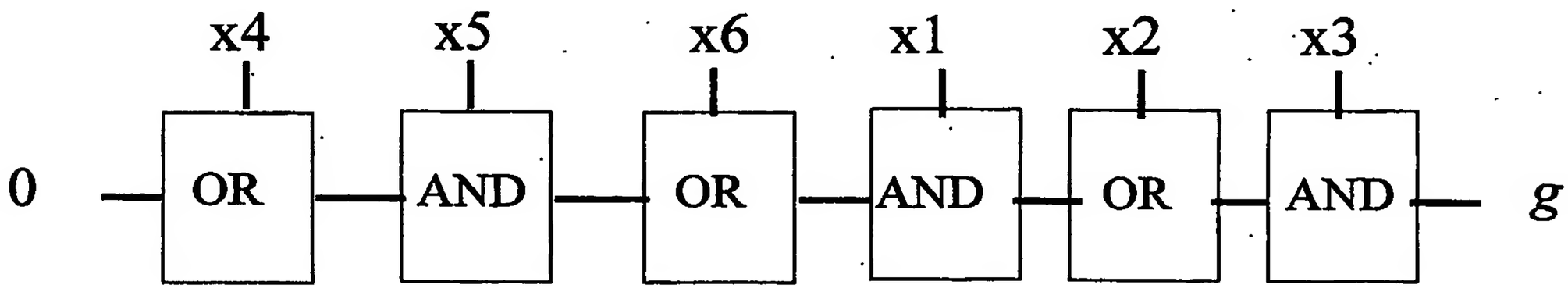
$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$



(*x1,x2,x3,x4,x5,x6*) is the optimal ordering for this function.

第４９（ｂ）図

$$g = ((x_4x_5 \vee x_6)x_1 \vee x_2)x_3$$

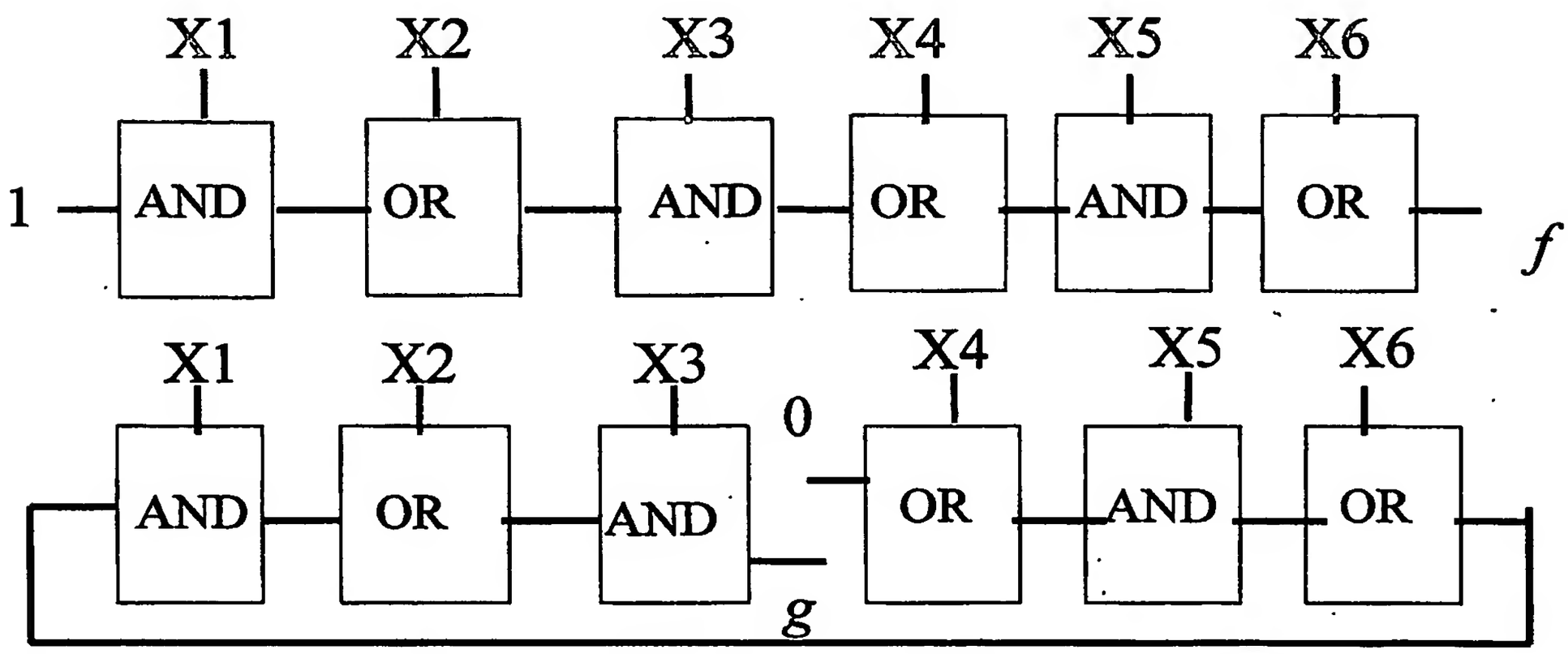


(*x4,x5,x6,x1,x2,x3*) is the optimal ordering for this function.

第５０図

$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$

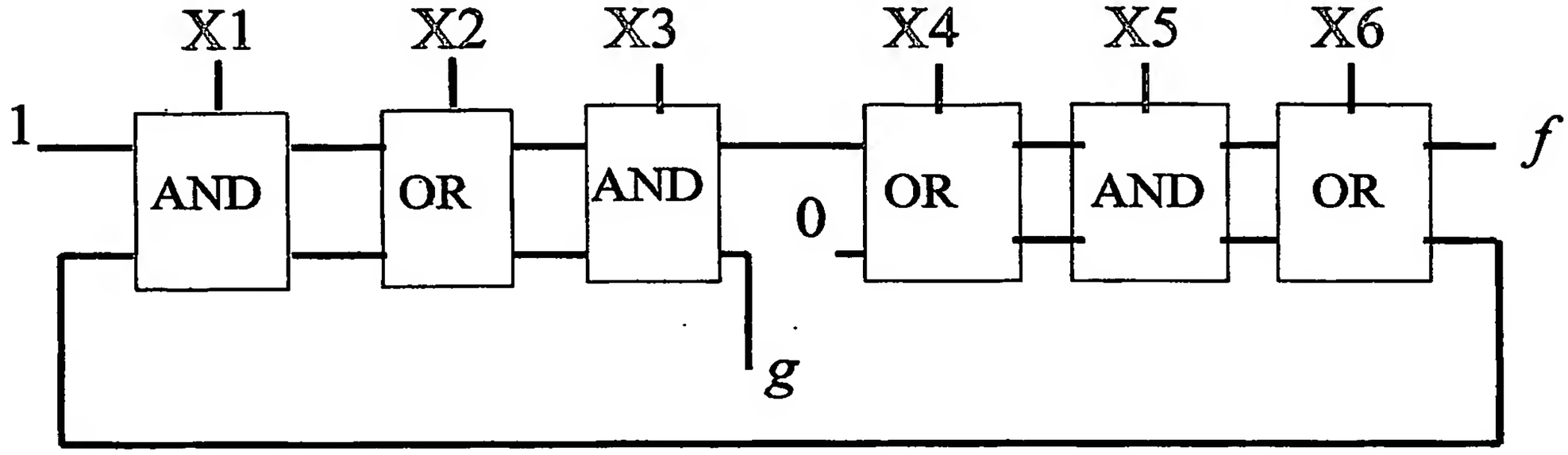
$$g = ((x_4 x_5 \vee x_6)x_1 \vee x_2)x_3$$



第５１図

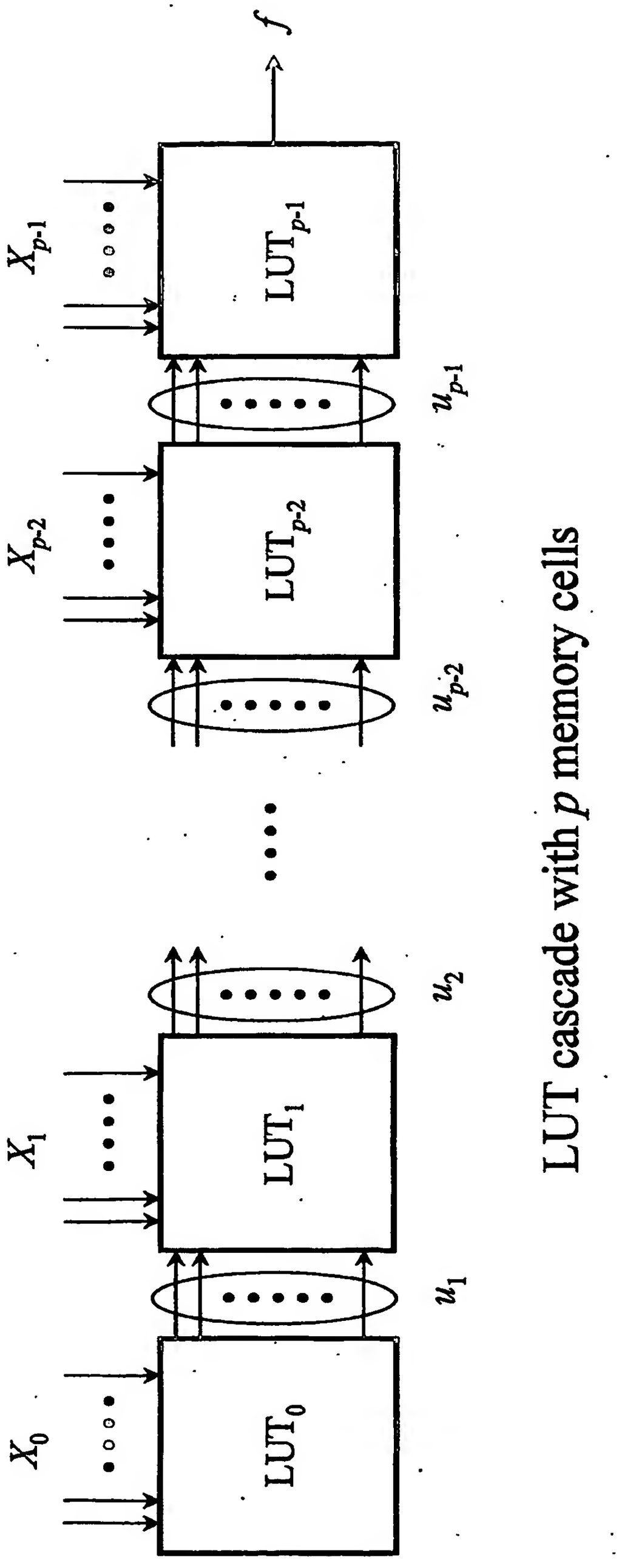
$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$

$$g = ((x_4 x_5 \vee x_6)x_1 \vee x_2)x_3$$



By using the LUT ring, the ordering can be optimal for two functions.

第５２図



LUT cascade with p memory cells

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/004752

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] G06F7/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] G06F7/00, H03K19/173

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2004
Kokai Jitsuyo Shinan Koho	1971-2004	Jitsuyo Shinan Toroku Koho	1996-2004

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Tsutomu SASAO et al., "Tashutsuryoku Kansu no Cascade Jutsugen to Saikosei Kano Hardware ni yoru Jitsugen", The Institute of Electronics, Information and Communication Engineers Gijutsu Kenkyu Hokoku, Vol.101, No.3, 06 April, 2001 (06.04.01), pages 57 to 64	1-13
A	Akihiko TOMITA et al., "LUT Array-gata PLD no Sekkei to Shisaku", The Institute of Electronics, Information and Communication Engineers Gijutsu Kenkyu Hokoku, Vol.100, No.475, 23 November, 2000 (23.11.00), pages 173 to 178	1-13

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 July, 2004 (22.07.04)	10 August, 2004 (10.08.04)
Name and mailing address of the ISA/ Japanese Patent Office	Authorized officer
Facsimile No.	Telephone No.

Form PCT/ISA/210 (second sheet) (January 2004)

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００４／００４７５２

A.　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Ｉｎｔ．Ｃｌ⁷　Ｇ０６Ｆ　７／００

B.　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Ｉｎｔ．Ｃｌ⁷　Ｇ０６Ｆ　７／００，　Ｈ０３Ｋ１９／１７３

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報　　１９２２－１９９６年
日本国公開実用新案公報　　１９７１－２００４年
日本国登録実用新案公報　　１９９４－２００４年
日本国実用新案登録公報　　１９９６－２００４年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C.　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	笹尾勤、他，多出力関数のカスケード実現と再構成可能ハードウェアによる実現，電子情報通信学会技術研究報告，Ｖｏｌ．１０１，Ｎｏ．３，２００１．０４．０６，Ｐ．５７－６４	１－１３
A	冨田明彦、他，ＬＵＴアレイ型ＰＬＤの設計と試作，電子情報通信学会技術研究報告，Ｖｏｌ．１００，Ｎｏ．４７５，２０００．１１．２３，Ｐ．１７３－１７８	１－１３

☐　Ｃ欄の続きにも文献が列挙されている。

☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
２２．０７．２００４	10.8.2004

国際調査機関の名称及びあて先	特許庁審査官（権限のある職員）		
日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	田中　友章	５Ｅ	９３７６
	電話番号　０３－３５８１－１１０１　内線　３５２０		